Red Hat Cluster Suite

Configuring and Managing a Cluster



Red Hat Cluster Suite: Configuring and Managing a Cluster

Copyright © 2000-2004 Red Hat, Inc.Mission Critical Linux, Inc.K.M. Sorenson



Red Hat, Inc.

1801 Varsity Drive
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

rh-cs(EN)-3-Print-RHI (2006-02-03T16:44)

For Part I *Using the Red Hat Cluster Manager* and Part III *Appendixes*, permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation. A copy of the license is available at http://www.gnu.org/licenses/fdl.html. The content described in this paragraph is copyrighted by © Mission Critical Linux, Inc. (2000), K.M. Sorenson (2000), and Red Hat, Inc. (2000-2003).
This material in Part II *Configuring a Linux Virtual Server Cluster* may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/). Distribution of substantively modified versions of this material is prohibited without the explicit permission of the copyright holder. Distribution of the work or derivative of the work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder. The content described in this paragraph is copyrighted by © Red Hat, Inc. (2000-2003).
Red Hat and the Red Hat "Shadow Man" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.
All other trademarks referenced herein are the property of their respective owners.
The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Table of Contents

Acknowledgments ..... **i**
Introduction ..... **iii**
1. How To Use This Manual iii
2. Document Conventions iv
3. More to Come vi
3.1. Send in Your Feedback vi
4. Activate Your Subscription vii
4.1. Provide a Red Hat Login vii
4.2. Provide Your Subscription Number vii
4.3. Connect Your System viii
I. Using the Red Hat Cluster Manager ..... **ix**
1. Red Hat Cluster Manager Overview 1
1.1. Red Hat Cluster Manager Features 2
2. Hardware Installation and Operating System Configuration 5
2.1. Choosing a Hardware Configuration 5
2.2. Setting Up the Members 15
2.3. Installing and Configuring Red Hat Enterprise Linux 17
2.4. Setting Up and Connecting the Cluster Hardware 21
3. Cluster Configuration 35
3.1. Installing the Red Hat Cluster Suite Packages 35
3.2. Installation Notes for Red Hat Enterprise Linux 2.1 Users 37
3.3. The **Cluster Configuration Tool** 37
3.4. Configuring the Cluster Software 40
3.5. Editing the `rawdevices` File 41
3.6. Configuring Cluster Daemons 42
3.7. Adding and Deleting Members 46
3.8. Configuring a Power Controller Connection 48
3.9. Configuring a Failover Domain 49
3.10. Adding a Service to the Cluster 51
3.11. Checking the Cluster Configuration 54
3.12. Configuring `syslogd` Event Logging 56
4. Service Administration 59
4.1. Configuring a Service 59
4.2. Displaying a Service Configuration 62
4.3. Disabling a Service 63
4.4. Enabling a Service 63
4.5. Modifying a Service 63
4.6. Relocating a Service 64
4.7. Deleting a Service 64
4.8. Handling Failed Services 64
5. Database Services 67
5.1. Setting Up an Oracle Service 67
5.2. Tuning Oracle Service 72
5.3. Setting Up a MySQL Service 73
6. Network File Sharing Services 77
6.1. Setting Up an NFS Service 77
6.2. Using the **NFS Druid** 77
6.3. NFS Caveats 82
6.4. Importing the Contents of an NFS Exports File 82
6.5. NFS Configuration: Active-Active Example 83
6.6. Setting Up a Samba Service 84
6.7. Using the Samba Druid 86
6.8. Fields in the `smb.conf.`*`sharename`* File 90

7. Setting Up Apache HTTP Server 93
7.1. Apache HTTP Server Setup Overview 93
7.2. Configuring Shared Storage 93
7.3. Installing and Configuring the Apache HTTP Server 94
8. Cluster Administration 97
8.1. Overview of the **Cluster Status Tool** 97
8.2. Displaying Cluster and Service Status 97
8.3. Starting and Stopping the Cluster Software 99
8.4. Modifying the Cluster Configuration 100
8.5. Backing Up and Restoring the Cluster Database 100
8.6. Modifying Cluster Event Logging 101
8.7. Updating the Cluster Software 101
8.8. Changing the Cluster Name 102
8.9. Disabling the Cluster Software 102
8.10. Diagnosing and Correcting Problems in a Cluster 102
II. Configuring a Linux Virtual Server Cluster 107
9. Introduction to Linux Virtual Server 109
9.1. Technology Overview 109
9.2. Basic Configurations 109
10. Linux Virtual Server Overview 111
10.1. A Basic LVS Configuration 111
10.2. A Three Tiered LVS Configuration 113
10.3. LVS Scheduling Overview 114
10.4. Routing Methods 116
10.5. Persistence and Firewall Marks 118
10.6. LVS Cluster — A Block Diagram 118
11. Initial LVS Configuration 121
11.1. Configuring Services on the LVS Routers 121
11.2. Setting a Password for the **Piranha Configuration Tool** 122
11.3. Starting the **Piranha Configuration Tool** Service 122
11.4. Limiting Access To the **Piranha Configuration Tool** 123
11.5. Turning on Packet Forwarding 124
11.6. Configuring Services on the Real Servers 124
12. Setting Up a Red Hat Enterprise Linux LVS Cluster 125
12.1. The NAT LVS Cluster 125
12.2. Putting the Cluster Together 127
12.3. Multi-port Services and LVS Clustering 128
12.4. FTP In an LVS Cluster 130
12.5. Saving Network Packet Filter Settings 132
13. Configuring the LVS Routers with **Piranha Configuration Tool** 133
13.1. Necessary Software 133
13.2. Logging Into the **Piranha Configuration Tool** 133
13.3. **CONTROL/MONITORING** 134
13.4. **GLOBAL SETTINGS** 135
13.5. **REDUNDANCY** 137
13.6. **VIRTUAL SERVERS** 139
13.7. Synchronizing Configuration Files 147
13.8. Starting the Cluster 148

III. Appendixes **149**
A. Using Red Hat Cluster Manager with Piranha 151
B. Using Red Hat GFS with Red Hat Cluster Suite 153
B.1. Terminology 153
B.2. Changes to Red Hat Cluster 154
B.3. Installation Scenarios 154
C. The GFS Setup Druid 157
C.1. Cluster Name 157
C.2. *LOCK_GULM* parameters 157
C.3. Choose Location for CCS Files 158
C.4. Cluster Members 159
C.5. Saving Your Configuration and Next Steps 161
D. Supplementary Hardware Information 163
D.1. Setting Up Power Controllers 163
D.2. SCSI Bus Configuration Requirements 165
D.3. SCSI Bus Termination 166
D.4. SCSI Bus Length 166
D.5. SCSI Identification Numbers 167
E. Supplementary Software Information 169
E.1. Cluster Communication Mechanisms 169
E.2. Failover and Recovery Scenarios 170
E.3. Common Cluster Behaviors: General 170
E.4. Common Behaviors: Two Member Cluster with Disk-based Tie-breaker 172
E.5. Common Behaviors: 2-4 Member Cluster with IP-based Tie-Breaker 173
E.6. Common Behaviors: 3-5 Member Cluster 173
E.7. Common Behaviors: Cluster Service Daemons 174
E.8. Common Behaviors: Miscellaneous 175
E.9. The `cluster.xml` File 175
F. Cluster Command-line Utilities 179
F.1. Using `redhat-config-cluster-cmd` 179
F.2. Using the `shutil` Utility 180
F.3. Using the `clusvcadm` Utility 180
F.4. Using the `clufence` Utility 181
Index **183**
Colophon **191**

Acknowledgments

The Red Hat Cluster Manager software was originally based on the open source Kimberlite (http://oss.missioncriticallinux.com/kimberlite/) cluster project, which was developed by Mission Critical Linux, Inc.

Subsequent to its inception based on Kimberlite, developers at Red Hat have made a large number of enhancements and modifications. The following is a non-comprehensive list highlighting some of these enhancements.

- Packaging and integration into the Red Hat installation paradigm to simplify the end user's experience.
- Addition of support for multiple cluster members.
- Addition of support for high availability NFS services.
- Addition of support for high availability Samba services.
- Addition of the **Cluster Configuration Tool**, a graphical configuration tool.
- Addition of the **Cluster Status Tool**, a graphical monitoring and administration tool.
- Addition of support for failover domains.
- Addition of support for Red Hat GFS, including the **GFS Setup Druid** and the **LOCK_GULM** fencing driver.
- Addition of support for using watchdog timers as a data integrity provision.
- Addition of service monitoring which automatically restart a failed application.
- Rewrite of the service manager to facilitate additional cluster-wide operations.
- A set of miscellaneous bug fixes.

The Red Hat Cluster Manager software incorporates STONITH compliant power switch modules from the Linux-HA project; refer to http://www.linux-ha.org/stonith/.

Introduction

The Red Hat Cluster Suite is a collection of technologies working together to provide data integrity and the ability to maintain application availability in the event of a failure. Administrators can deploy enterprise cluster solutions using a combination of hardware redundancy along with the failover and load-balancing technologies in Red Hat Cluster Suite.

Red Hat Cluster Manager is a high-availability cluster solution specifically suited for database applications, network file servers, and World Wide Web (Web) servers with dynamic content. An Red Hat Cluster Manager system features data integrity and application availability using redundant hardware, shared disk storage, power management, and robust cluster communication and application failover mechanisms.

Administrators can also deploy highly available applications services using Piranha, a load-balancing and advanced routing cluster solution based on Linux Virtual Server (LVS) technology. Using Piranha, administrators can build highly available e-commerce sites that feature complete data integrity and service availability, in addition to load balancing capabilities. Refer to Part II *Configuring a Linux Virtual Server Cluster* for more information.

This guide assumes that the user has an advanced working knowledge of Red Hat Enterprise Linux and understands the concepts of server computing. For more information about using Red Hat Enterprise Linux, refer to the following resources:

- *Red Hat Enterprise Linux Installation Guide* for information regarding installation.
- *Red Hat Enterprise Linux Introduction to System Administration* for introductory information for new Red Hat Enterprise Linux system administrators.
- *Red Hat Enterprise Linux System Administration Guide* for more detailed information about configuring Red Hat Enterprise Linux to suit your particular needs as a user.
- *Red Hat Enterprise Linux Reference Guide* provides detailed information suited for more experienced users to refer to when needed, as opposed to step-by-step instructions.
- *Red Hat Enterprise Linux Security Guide* details the planning and the tools involved in creating a secured computing environment for the data center, workplace, and home.

HTML, PDF, and RPM versions of the manuals are available on the Red Hat Enterprise Linux Documentation CD and online at:

`http://www.redhat.com/docs/`

1. How To Use This Manual

This manual contains information about setting up a Red Hat Cluster Manager system. These tasks are described in Chapter 2 *Hardware Installation and Operating System Configuration* and Chapter 3 *Cluster Configuration*.

For information about configuring Red Hat Cluster Manager cluster services, refer to Chapter 4 *Service Administration* through Chapter 7 *Setting Up Apache HTTP Server*.

Part II *Configuring a Linux Virtual Server Cluster* describes how to achieve load balancing in an Red Hat Enterprise Linux cluster by using the Linux Virtual Server.

For information about deploying a Red Hat Cluster Manager cluster in conjunction with Red Hat GFS using the **GFS Setup Druid**, refer to Appendix C *The GFS Setup Druid*.

Appendix D *Supplementary Hardware Information* contains detailed configuration information on specific hardware devices and shared storage configurations. Appendix E *Supplementary Software Information* contains background information on the cluster software and other related information.

Appendix F *Cluster Command-line Utilities* provides usage and reference information on the command-line utilities included with Red Hat Cluster Suite.

This guide assumes you have a thorough understanding of Red Hat Enterprise Linux system administration concepts and tasks. For detailed information on Red Hat Enterprise Linux system administration, refer to the *Red Hat Enterprise Linux System Administration Guide*. For reference information on Red Hat Enterprise Linux, refer to the *Red Hat Enterprise Linux Reference Guide*.

2. Document Conventions

When you read this manual, certain words are represented in different fonts, typefaces, sizes, and weights. This highlighting is systematic; different words are represented in the same style to indicate their inclusion in a specific category. The types of words that are represented this way include the following:

`command`

Linux commands (and other operating system commands, when used) are represented this way. This style should indicate to you that you can type the word or phrase on the command line and press [Enter] to invoke a command. Sometimes a command contains words that would be displayed in a different style on their own (such as file names). In these cases, they are considered to be part of the command, so the entire phrase is displayed as a command. For example:

Use the `cat testfile` command to view the contents of a file, named `testfile`, in the current working directory.

`file name`

File names, directory names, paths, and RPM package names are represented this way. This style should indicate that a particular file or directory exists by that name on your system. Examples:

The `.bashrc` file in your home directory contains bash shell definitions and aliases for your own use.

The `/etc/fstab` file contains information about different system devices and file systems.

Install the `webalizer` RPM if you want to use a Web server log file analysis program.

application

This style indicates that the program is an end-user application (as opposed to system software). For example:

Use **Mozilla** to browse the Web.

[key]

A key on the keyboard is shown in this style. For example:

To use [Tab] completion, type in a character and then press the [Tab] key. Your terminal displays the list of files in the directory that start with that letter.

[key]-[combination]

A combination of keystrokes is represented in this way. For example:

The [Ctrl]-[Alt]-[Backspace] key combination exits your graphical session and return you to the graphical login screen or the console.

text found on a GUI interface

A title, word, or phrase found on a GUI interface screen or window is shown in this style. Text shown in this style is being used to identify a particular GUI screen or an element on a GUI screen (such as text associated with a checkbox or field). Example:

Select the **Require Password** checkbox if you would like your screensaver to require a password before stopping.

top level of a menu on a GUI screen or window

A word in this style indicates that the word is the top level of a pulldown menu. If you click on the word on the GUI screen, the rest of the menu should appear. For example:

Under **File** on a GNOME terminal, the **New Tab** option allows you to open multiple shell prompts in the same window.

If you need to type in a sequence of commands from a GUI menu, they are shown like the following example:

Go to **Main Menu Button** (on the Panel) => **Programming** => **Emacs** to start the **Emacs** text editor.

button on a GUI screen or window

This style indicates that the text can be found on a clickable button on a GUI screen. For example:

Click on the **Back** button to return to the webpage you last viewed.

`computer output`

Text in this style indicates text displayed to a shell prompt such as error messages and responses to commands. For example:

The `ls` command displays the contents of a directory. For example:

```
Desktop                about.html       logs           paulwesterberg.png
Mail                   backupfiles      mail           reports
```

The output returned in response to the command (in this case, the contents of the directory) is shown in this style.

`prompt`

A prompt, which is a computer's way of signifying that it is ready for you to input something, is shown in this style. Examples:

```
$
```

```
#
```

```
[stephen@maturin stephen]$
```

```
leopard login:
```

`user input`

Text that the user has to type, either on the command line, or into a text box on a GUI screen, is displayed in this style. In the following example, **`text`** is displayed in this style:

To boot your system into the text based installation program, you must type in the **`text`** command at the `boot:` prompt.

`replaceable`

Text used for examples, which is meant to be replaced with data provided by the user, is displayed in this style. In the following example, *<version-number>* is displayed in this style:

The directory for the kernel source is `/usr/src/<version-number>/`, where `<version-number>` is the version of the kernel installed on this system.

Additionally, we use several different strategies to draw your attention to certain pieces of information. In order of how critical the information is to your system, these items are marked as a note, tip, important, caution, or warning. For example:



Note

Remember that Linux is case sensitive. In other words, a rose is not a ROSE is not a rOsE.



Tip

The directory `/usr/share/doc/` contains additional documentation for packages installed on your system.



Important

If you modify the DHCP configuration file, the changes do not take effect until you restart the DHCP daemon.



Caution

Do not perform routine tasks as root — use a regular user account unless you need to use the root account for system administration tasks.



Warning

Be careful to remove only the necessary Red Hat Enterprise Linux partitions. Removing other partitions could result in data loss or a corrupted system environment.

3. More to Come

This manual is part of Red Hat's growing commitment to provide useful and timely support to Red Hat Enterprise Linux users.

3.1. Send in Your Feedback

If you spot a typo, or if you have thought of a way to make this manual better, we would love to hear from you. Please submit a report in Bugzilla (http://bugzilla.redhat.com/bugzilla/) against the component `rh-cs`.

Be sure to mention the manual's identifier:

```
rh-cs(EN)-3-Print-RHI (2006-02-03T16:44)
```

By mentioning this manual's identifier, we know exactly which version of the guide you have.

If you have a suggestion for improving the documentation, try to be as specific as possible. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

4. Activate Your Subscription

Before you can access service and software maintenance information, and the support documentation included in your subscription, you must activate your subscription by registering with Red Hat. Registration includes these simple steps:

- Provide a Red Hat login
- Provide a subscription number
- Connect your system

The first time you boot your installation of Red Hat Enterprise Linux, you are prompted to register with Red Hat using the **Setup Agent**. If you follow the prompts during the **Setup Agent**, you can complete the registration steps and activate your subscription.

If you can not complete registration during the **Setup Agent** (which requires network access), you can alternatively complete the Red Hat registration process online at http://www.redhat.com/register/.

4.1. Provide a Red Hat Login

If you do not have an existing Red Hat login, you can create one when prompted during the **Setup Agent** or online at:

```
https://www.redhat.com/apps/activate/newlogin.html
```

A Red Hat login enables your access to:

- Software updates, errata and maintenance via Red Hat Network
- Red Hat technical support resources, documentation, and Knowledgebase

If you have forgotten your Red Hat login, you can search for your Red Hat login online at:

```
https://rhn.redhat.com/help/forgot_password.pxt
```

4.2. Provide Your Subscription Number

Your subscription number is located in the package that came with your order. If your package did not include a subscription number, your subscription was activated for you and you can skip this step.

You can provide your subscription number when prompted during the **Setup Agent** or by visiting http://www.redhat.com/register/.

4.3. Connect Your System

The Red Hat Network Registration Client helps you connect your system so that you can begin to get updates and perform systems management. There are three ways to connect:

1. During the **Setup Agent** — Check the **Send hardware information** and **Send system package list** options when prompted.
2. After the **Setup Agent** has been completed — From the **Main Menu**, go to **System Tools**, then select **Red Hat Network**.
3. After the **Setup Agent** has been completed — Enter the following command from the command line as the root user:
   - `/usr/bin/up2date --register`

I. Using the Red Hat Cluster Manager

Clustered systems provide reliability, scalability, and availability to critical production services. Using the Red Hat Cluster Manager, administrators can create high availability clusters for filesharing, Web servers, databases, and more. This part discusses the installation and configuration of cluster systems using the recommended hardware and Red Hat Enterprise Linux.

This section is licensed under the GNU Free Documentation License. For details refer to the Copyright page.

Table of Contents

1. Red Hat Cluster Manager Overview 1
2. Hardware Installation and Operating System Configuration 5
3. Cluster Configuration 35
4. Service Administration 59
5. Database Services 67
6. Network File Sharing Services 77
7. Setting Up Apache HTTP Server 93
8. Cluster Administration 97

Chapter 1.

Red Hat Cluster Manager Overview

Red Hat Cluster Manager allows administrators to connect separate systems (called members or nodes) together to create failover clusters that ensure application availability and data integrity under several failure conditions. Administrators can use Red Hat Cluster Manager with database applications, file sharing services, web servers, and more.

To set up a failover cluster, you must connect the *member systems* (often referred to simply as *members* or *nodes*) to the cluster hardware, and configure the members into the cluster environment. The foundation of a cluster is an advanced host membership algorithm. This algorithm ensures that the cluster maintains complete data integrity at all times by using the following methods of inter-member communication:

- Network connections between the cluster systems for *heartbeat*
- *Shared state* on shared disk storage to hold cluster status

To make an application and data highly available in a cluster, you must configure a *service* (such as an application and shared disk storage) as a discrete, named group of properties and resources to which you can assign an IP address to provide transparent client access. For example, you can set up a service that provides clients with access to highly-available database application data.

You can associate a service with a *failover domain*, a subset of cluster members that are eligible to run the service. In general, any eligible member can run the service and access the service data on shared disk storage. However, each service can run on only one cluster member at a time, in order to maintain data integrity. You can specify whether or not the members in a failover domain are ordered by preference. You can also specify whether or not a service is restricted to run only on members of its associated failover domain. (When associated with an unrestricted failover domain, a service can be started on any cluster member in the event no member of the failover domain is available.)

You can set up an *active-active* configuration in which the members run different services simultaneously, or a *hot-standby* configuration in which a primary member runs all the services, and a backup cluster system takes over only if the primary system fails.

Figure 1-1 shows an example of a cluster in an active-active configuration.

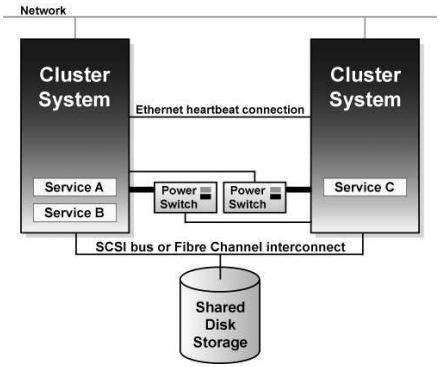


Figure 1-1. Example Cluster in Active-Active Configuration

If a hardware or software failure occurs, the cluster automatically restarts the failed member's services on the functional member. This *service failover* capability ensures that no data is lost, and there is little disruption to users. When the failed member recovers, the cluster can re-balance the services across the members.

In addition, you can cleanly stop the services running on a cluster system and then restart them on another system. This *service relocation* capability allows you to maintain application and data availability when a cluster member requires maintenance.

1.1. Red Hat Cluster Manager Features

Cluster systems deployed with Red Hat Cluster Manager include the following features:

No-single-point-of-failure hardware configuration

Clusters can include a dual-controller RAID array, multiple network channels, and redundant uninterruptible power supply (UPS) systems to ensure that no single failure results in application down time or loss of data.

Alternately, a low-cost cluster can be set up to provide less availability than a no-single-point-of-failure cluster. For example, you can set up a cluster with a single-controller RAID array and only a single Ethernet channel.

Certain low-cost alternatives, such as software RAID and multi-initiator parallel SCSI, are not compatible or appropriate for use on the shared cluster storage. Refer to Section 2.1 Choosing a Hardware Configuration, for more information.

Service configuration framework

Clusters allow you to easily configure individual services to make data and applications highly available. To create a service, you specify the resources used in the service and properties for the service, including the service name, application start, stop, and status script, disk partitions, mount points, and the cluster members on which you prefer the service to run. After you add a service, the cluster management software stores the information in a cluster configuration file on shared storage, where the configuration data can be accessed by all cluster members.

The cluster provides an easy-to-use framework for database applications. For example, a database service serves highly-available data to a database application. The application running on a cluster member provides network access to database client systems, such as Web servers. If the service fails over to another member, the application can still access the shared database data. A network-accessible database service is usually assigned an IP address, which is failed over along with the service to maintain transparent access for clients.

The cluster service framework can be easily extended to other applications, as well.

Failover domains

By assigning a service to a *restricted failover domain*, you can limit the members that are eligible to run a service in the event of a failover. (A service that is assigned to a restricted failover domain cannot be started on a cluster member that is not included in that failover domain.) You can order the members in a failover domain by preference to ensure that a particular member runs the service (as long as that member is active). If a service is assigned to an unrestricted failover domain, the service starts on any available cluster member (if none of the members of the failover domain are available).

Data integrity assurance

To ensure data integrity, only one member can run a service and access service data at one time. The use of power switches in the cluster hardware configuration enables a member to power-cycle another member before restarting that member's services during the failover process.

This prevents any two systems from simultaneously accessing the same data and corrupting it. Although not required, it is recommended that power switches are used to guarantee data integrity under all failure conditions. Watchdog timers are an optional variety of power control to ensure correct operation of service failover.

Cluster administration user interface

The cluster administration interface facilitiates management tasks such as: creating, starting, and stopping services; relocating services from one member to another; modifying the cluster configuration (to add or remove services or resources); and monitoring the cluster members and services.

Ethernet channel bonding

To monitor the health of the other members, each member monitors the health of the remote power switch, if any, and issues heartbeat pings over network channels. With Ethernet channel bonding, multiple Ethernet interfaces are configured to behave as one, reducing the risk of a single-point-of-failure in the typical switched Ethernet connection between systems.

Shared storage for quorum information

Shared state information includes whether the member is active. Service state information includes whether the service is running and which member is running the service. Each member checks to ensure that the status of the other members is up to date.

In a two-member cluster, each member periodically writes a timestamp and cluster state information to two shared cluster partitions located on shared disk storage. To ensure correct cluster operation, if a member is unable to write to both the primary and shadow shared cluster partitions at startup time, it is not allowed to join the cluster. In addition, if a member is not updating its timestamp, and if heartbeats to the system fail, the member is removed from the cluster.

Figure 1-2 shows how members communicate in a cluster configuration. Note that the terminal server used to access system consoles via serial ports is not a required cluster component.

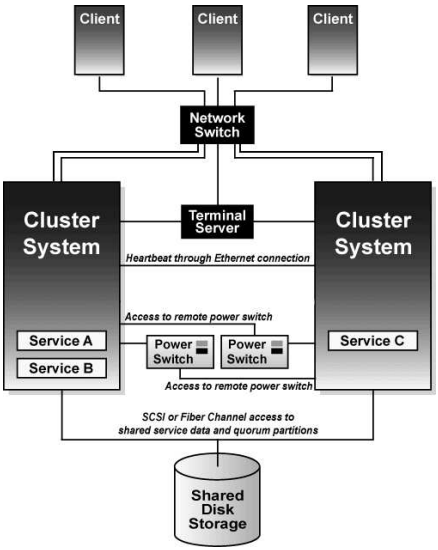


Figure 1-2. Cluster Communication Mechanisms

Service failover capability

If a hardware or software failure occurs, the cluster takes the appropriate action to maintain application availability and data integrity. For example, if a member completely fails, another member (in the associated failover domain, if used, or in the cluster) restarts its services. Services already running on this member are not disrupted.

When the failed member reboots and is able to write to the shared cluster partitions, it can rejoin the cluster and run services. Depending on how the services are configured, the cluster can rebalance the services among the members.

Manual service relocation capability

In addition to automatic service failover, a cluster allows you to cleanly stop services on one member and restart them on another member. You can perform planned maintenance on a member system while continuing to provide application and data availability.

Event logging facility

To ensure that problems are detected and resolved before they affect service availability, the cluster daemons log messages by using the conventional Linux syslog subsystem. You can customize the severity level of the logged messages.

Application monitoring

The infrastructure in a cluster can optionally monitor the state and health of an application. In this manner, should an application-specific failure occur, the cluster automatically restarts the application. In response to the application failure, the application attempts to be restarted on the member it was initially running on; failing that, it restarts on another cluster member. You can specify which members are eligible to run a service by assigning a failover domain to the service.

Chapter 2.

Hardware Installation and Operating System Configuration

To set up the hardware configuration and install Red Hat Enterprise Linux, follow these steps:

- Choose a cluster hardware configuration that meets the needs of applications and users; refer to Section 2.1 *Choosing a Hardware Configuration*.
- Set up and connect the members and the optional console switch and network switch or hub; refer to Section 2.2 *Setting Up the Members*.
- Install and configure Red Hat Enterprise Linux on the cluster members; refer to Section 2.3 *Installing and Configuring Red Hat Enterprise Linux*.
- Set up the remaining cluster hardware components and connect them to the members; refer to Section 2.4 *Setting Up and Connecting the Cluster Hardware*.

After setting up the hardware configuration and installing Red Hat Enterprise Linux, install the cluster software.



Tip

Refer to the *Red Hat Hardware Compatibility List* available at http://hardware.redhat.com/hcl/ for a list of compatible hardware. Perform a **Quick Search** for the term `cluster` to find results for power switch and shared storage hardware certified for or compatible with Red Hat Cluster Manager. For general system hardware compatibility searches, use manufacturer, brand, and/or model keywords to check for compatibility with Red Hat Enterprise Linux.

2.1. Choosing a Hardware Configuration

The Red Hat Cluster Manager allows administrators to use commodity hardware to set up a cluster configuration that meets the performance, availability, and data integrity needs of applications and users. Cluster hardware ranges from low-cost minimum configurations that include only the components required for cluster operation, to high-end configurations that include redundant Ethernet channels, hardware RAID, and power switches.

Regardless of configuration, the use of high-quality hardware in a cluster is recommended, as hardware malfunction is a primary cause of system down time.

Although all cluster configurations provide availability, some configurations protect against every *single point of failure*. In addition, all cluster configurations provide data integrity, but some configurations protect data under every failure condition. Therefore, administrators must fully understand the needs of their computing environment and also the availability and data integrity features of different hardware configurations to choose the cluster hardware that meets the proper requirements.

When choosing a cluster hardware configuration, consider the following:

Performance requirements of applications and users

Choose a hardware configuration that provides adequate memory, CPU, and I/O resources. Be sure that the configuration chosen can handle any future increases in workload as well.

Cost restrictions

The hardware configuration chosen must meet budget requirements. For example, systems with multiple I/O ports usually cost more than low-end systems with fewer expansion capabilities.

Availability requirements

If a computing environment requires the highest degree of availability, such as a production environment, then a cluster hardware configuration that protects against all single points of failure, including disk, storage interconnect, Ethernet channel, and power failures is recommended. Environments that can tolerate an interruption in availability, such as development environments, may not require as much protection. Refer to Section 2.4.3 *Configuring UPS Systems* and Section 2.4.4 *Configuring Shared Disk Storage* for more information about using redundant hardware for high availability.

Data integrity under all failure conditions requirement

Using power switches in a cluster configuration guarantees that service data is protected under every failure condition. These devices enable a member to power cycle another member before restarting its services during failover. Power switches protect against data corruption if an unresponsive (or hanging) member becomes responsive after its services have failed over and then issues I/O to a disk that is also receiving I/O from the other member.

In addition, if a quorum daemon fails on a member, the member is no longer able to monitor the shared cluster partitions. If you are not using power switches in the cluster, this error condition may result in services being run on more than one member, which can cause data corruption. Refer to Section 2.4.2 *Configuring Power Switches* for more information about the benefits of using power switches in a cluster. It is recommended that production environments use power switches or watchdog timers in the cluster configuration.

2.1.1. Shared Storage Requirements

The operation of the cluster depends on reliable, coordinated access to shared storage. In the event of hardware failure, it is desirable to be able to disconnect one member from the shared storage for repair without disrupting the other members. Shared storage is truly vital to the cluster configuration.

Testing has shown that it is difficult, if not impossible, to configure reliable multi-initiator parallel SCSI configurations at data rates above 80MB/sec using standard SCSI adapters. Further tests have shown that these configurations cannot support online repair because the bus does not work reliably when the HBA terminators are disabled, and external terminators are used. For these reasons, multi-initiator SCSI configurations using standard adapters are not supported. Either single-initiator SCSI bus adapters (connected to multi-ported storage) or Fibre Channel adapters are required.

The Red Hat Cluster Manager requires that all cluster members have simultaneous access to the shared storage. Certain host RAID adapters are capable of providing this type of access to shared RAID units. These products require extensive testing to ensure reliable operation, especially if the shared RAID units are based on parallel SCSI buses. These products typically do not allow for online repair of a failed member. Only host RAID adapters listed in the *Red Hat Hardware Compatibility List* are supported.

The use of software RAID, or software Logical Volume Management (LVM), is not supported on shared storage. This is because these products do not coordinate access from multiple hosts to shared storage. Software RAID or LVM may be used on non-shared storage on cluster members (for example, boot and system partitions, and other file systems which are not associated with any cluster services).

2.1.2. Minimum Hardware Requirements

A *minimum hardware configuration* includes only the hardware components that are required for cluster operation, as follows:

- Two servers to run cluster services
- Ethernet connection for sending heartbeat pings and for client network access
- Shared disk storage for the shared cluster partitions and service data

The hardware components described in Table 2-1 can be used to set up a minimum cluster configuration. This configuration does not guarantee data integrity under all failure conditions, because it does not include power switches. Note that this is a sample configuration; it is possible to set up a minimum configuration using other hardware.



Warning

The minimum cluster configuration is not a supported solution and *should not be used* in a production environment, as it does not guarantee data integrity under all failure conditions.

Hardware	Description
Two servers	Each member includes a network interface for client access and for Ethernet connections and a SCSI adapter (termination disabled) for the shared storage connection
Two network cables with RJ45 connectors	Network cables connect an Ethernet network interface on each member to the network for client access and heartbeat pings.
RAID storage enclosure	The RAID storage enclosure contains one controller with at least two host ports.
Two HD68 SCSI cables	Each cable connects one HBA to one port on the RAID controller, creating two single-initiator SCSI buses.

Table 2-1. Example of Minimum Cluster Configuration

The minimum hardware configuration is the most cost-effective cluster configuration; however, it includes multiple points of failure. For example, if the RAID controller fails, then all cluster services become unavailable. When deploying the minimal hardware configuration, software watchdog timers should be configured as a data integrity provision. Refer to Section D.1.2.3 *Configuring a Hardware Watchdog Timer* for details.

To improve availability, protect against component failure, and guarantee data integrity under all failure conditions, the minimum configuration can be expanded, as described in Table 2-2.

Problem	Solution
Disk failure	Hardware RAID to replicate data across multiple disks
RAID controller failure	Dual RAID controllers to provide redundant access to disk data
Heartbeat failure	Ethernet channel bonding and failover
Power source failure	Redundant uninterruptible power supply (UPS) systems
Data corruption under all failure conditions	Power switches or hardware-based watchdog timers

Table 2-2. Improving Availability and Guaranteeing Data Integrity

A no single point of failure hardware configuration that guarantees data integrity under all failure conditions can include the following components:

- At least two servers to run cluster services
- Ethernet connection between each member for heartbeat pings and for client network access
- Dual-controller RAID array to replicate shared partitions and service data
- Power switches to enable each member to power-cycle the other members during the failover process
- Ethernet interfaces configured to use channel bonding
- At least two UPS systems for a highly-available source of power

The components described in Table 2-3 can be used to set up a no single point of failure cluster configuration that includes two single-initiator SCSI buses and power switches to guarantee data integrity under all failure conditions. Note that this is a sample configuration; it is possible to set up a no single point of failure configuration using other hardware.

Hardware	Description
Two servers (up to 8 supported)	Each member includes the following hardware: Two network interfaces for: Point-to-point Ethernet connections Client network access and Ethernet heartbeat pings Three serial ports for: Remote power switch connection Connection to the terminal server One Adaptec 29160 adapter (termination enabled) for the shared disk storage connection.
One network switch	A network switch enables the connection of multiple members to a network.
One Cyclades terminal server	A terminal server allows for management of remote members from a central location. (A terminal server is not required for cluster operation.)
Four network cables	Network cables connect the terminal server and a network interface on each member to the network switch.
Two RJ45 to DB9 crossover cables	RJ45 to DB9 crossover cables connect a serial port on each member to the Cyclades terminal server.
Two serial-attached power switches	Power switches enable each member to power-cycle the other member before restarting its services. The power cable for each member is connected to its own power switch. Note that serial-attach power switches are supported in two-member clusters only.
Two null modem cables	Null modem cables connect a serial port on each member to the power switch that provides power to the other member. This connection enables each member to power-cycle the other member.
FlashDisk RAID Disk Array with dual controllers	Dual RAID controllers protect against disk and controller failure. The RAID controllers provide simultaneous access to all the logical units on the host ports.

Hardware	Description
Two HD68 SCSI cables	HD68 cables connect each host bus adapter to a RAID enclosure "in" port, creating two single-initiator SCSI buses.
Two terminators	Terminators connected to each "out" port on the RAID enclosure terminate both single-initiator SCSI buses.
Redundant UPS Systems	UPS systems provide a highly-available source of power. The power cables for the power switches and the RAID enclosure are connected to two UPS systems.

Table 2-3. Example of a No Single Point of Failure Configuration

Figure 2-1 shows an example of a no single point of failure hardware configuration that includes the previously-described hardware, two single-initiator SCSI buses, and power switches to guarantee data integrity under all error conditions. A "T" enclosed in a circle represents a SCSI terminator.

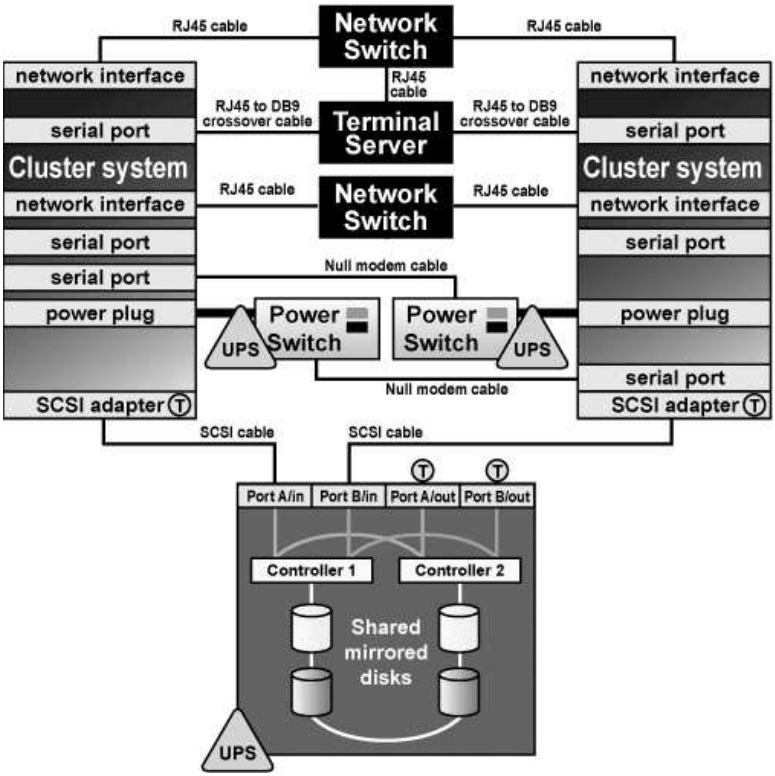


Figure 2-1. No Single Point of Failure Configuration Example

Cluster hardware configurations can also include other optional hardware components that are common in a computing environment. For example, a cluster can include a *network switch* or *network hub*, which enables the connection of the members to a network. A cluster may also include a *console*

switch, which facilitates the management of multiple members and eliminates the need for separate monitors, mouses, and keyboards for each member.

One type of console switch is a *terminal server*, which enables connection to serial consoles and management of many members from one remote location. As a low-cost alternative, you can use a *KVM* (keyboard, video, and mouse) switch, which enables multiple members to share one keyboard, monitor, and mouse. A KVM is suitable for configurations in which access to a graphical user interface (GUI) to perform system management tasks is preferred.

When choosing a system, be sure that it provides the required PCI slots, network slots, and serial ports. For example, a no single point of failure configuration requires multiple bonded Ethernet ports. Refer to Section 2.2.1 *Installing the Basic Cluster Hardware* for more information.

2.1.3. Choosing the Type of Power Controller

The Red Hat Cluster Manager implementation consists of a generic power management layer and a set of device-specific modules which accommodate a range of power management types. When selecting the appropriate type of power controller to deploy in the cluster, it is important to recognize the implications of specific device types. The following describes the types of supported power switches followed by a summary table. For a more detailed description of the role a power switch plays to ensure data integrity, refer to Section 2.4.2 *Configuring Power Switches*.

Serial-attached and network-attached power switches are separate devices which enable one cluster member to power cycle another member. They resemble a power plug strip on which individual outlets can be turned on and off under software control through either a serial or network cable. Network-attached power switches differ from serial-attached in that they connect to cluster members via an Ethernet hub or switch, rather than direct connection to cluster members. A network-attached power switch can not be directly attached to a cluster member using a crossover cable, as the power switch would be unable to power cycle the other members.

Watchdog timers provide a means for failed members to remove themselves from the cluster prior to another member taking over its services, rather than allowing one cluster member to power cycle another. The normal operational mode for watchdog timers is that the cluster software must periodically reset a timer prior to its expiration. If the cluster software fails to reset the timer, the watchdog triggers under the assumption that the member may have hung or otherwise failed. The healthy cluster member allows a window of time to pass prior to concluding that another cluster member has failed (by default, this window is 12 seconds). The watchdog timer interval must be less than the duration of time for one cluster member to conclude that another has failed. In this manner, a healthy member can assume, prior to taking over services, that the failed cluster member has safely removed itself from the cluster (by rebooting) and is no longer a risk to data integrity. The underlying watchdog support is included in the core Linux kernel. Red Hat Cluster Manager utilizes these watchdog features via its standard APIs and configuration mechanism.

There are two types of watchdog timers: hardware-based and software-based. Hardware-based watchdog timers typically consist of system board components such as the Intel® i810 TCO chipset. This circuitry has a high degree of independence from the main system CPU. This independence is beneficial in failure scenarios of a true system hang, as in this case it pulls down the system's reset lead resulting in a system reboot. Some PCI expansion cards provide watchdog features.

Software-based watchdog timers do not have any dedicated hardware. The implementation is a kernel thread which is periodically run; if the timer duration has expired, the thread initiates a system reboot. The vulnerability of the software watchdog timer is that under certain failure scenarios, such as system hangs while interrupts are blocked, the kernel thread is not called. As a result, in such conditions it cannot be definitively depended on for data integrity. This can cause the healthy cluster member to take over services for a hung member which could cause data corruption under certain scenarios.

Finally, administrators can choose not to employ a power controller at all. When a power controller is not in use, no provision exists for a cluster member to power cycle a failed member. Similarly, the failed member cannot be guaranteed to reboot itself under all failure conditions.



Important

Use of a power controller is *strongly* recommended as part of a production cluster environment. Configuration of a cluster without a power controller is not supported.

Ultimately, the right type of power controller deployed in a cluster environment depends on the data integrity requirements weighed against the cost and availability of external power switches.

Table 2-4 summarizes the types of supported power management modules and discusses their advantages and disadvantages individually.

Type	Notes	Pros	Cons
Serial-attached power switches (supported for two-member clusters only)	Two serial attached power controllers are used in a cluster (one per member)	Affords strong data integrity guarantees — the power controller itself is not a single point of failure as there are two in a cluster	Requires purchase of power controller hardware and cables; consumes serial ports; can only be used in two-member cluster
Network-attached power switches	A single network attached power controller is required per cluster (depending on the number of members); however, up to three are supported for each cluster member	Affords strong data integrity guarantees and can be used in clusters with more than two members	Requires purchase of power controller hardware — the power controller itself can become a single point of failure (although they are typically very reliable devices)
Hardware Watchdog Timer	Affords strong data integrity guarantees	Obviates the need to purchase external power controller hardware	Not all systems include supported watchdog hardware
Software Watchdog Timer	Offers acceptable data integrity provisions	Obviates the need to purchase external power controller hardware; works on any system	Under some failure scenarios, the software watchdog is not operational, opening a small vulnerability window
No power controller	No power controller function is in use	Obviates the need to purchase external power controller hardware; works on any system	Vulnerable to data corruption under certain failure scenarios

Table 2-4. Power Switches

2.1.4. Cluster Hardware Components

Use the following tables to identify the hardware components required for the cluster configuration.

Table 2-5 includes the hardware required for the cluster members.

Hardware	Quantity	Description	Required
Cluster members	eight (maximum supported)	Each member must provide enough PCI slots, network slots, and serial ports for the cluster hardware configuration. Because disk devices must have the same name on each member, it is recommended that the members have symmetric I/O subsystems. It is also recommended that the processor speed and amount of system memory be adequate for the processes run on the cluster members. Consult the *Red Hat Enterprise Linux 3 Release Notes* for specifics. Refer to Section 2.2.1 *Installing the Basic Cluster Hardware* for more information.	Yes

Table 2-5. Cluster Member Hardware

Table 2-6 includes several different types of power switches.

A single cluster requires only one type of power switch.

Hardware	Quantity	Description	Required
Serial power switches	Two	In a two-member cluster, use serial power switches to enable each cluster member to power-cycle the other member. Refer to Section 2.4.2 *Configuring Power Switches* for more information. Note, cluster members are configured with either serial power switches (supported for two-member clusters only) or network-attached power switches, but not both.	Strongly recommended for data integrity under all failure conditions
Null modem cable	Two	Null modem cables connect a serial port on a cluster member to a serial power switch. This enables each member to power-cycle the other member. Some power switches may require different cables.	Only if using serial power switches
Mounting bracket	One	Some power switches support rack mount configurations and require a separate mounting bracket.	Only for rack mounting power switches
Network power switch	One (depends on member count)	Network-attached power switches enable each cluster member to power cycle all others. Refer to Section 2.4.2 *Configuring Power Switches* for more information.	Strongly recommended for data integrity under all failure conditions
Watchdog Timer	One per member	Watchdog timers cause a failed cluster member to remove itself from a cluster prior to a healthy member taking over its services. Refer to Section 2.4.2 *Configuring Power Switches* for more information.	Recommended for data integrity on systems which provide integrated watchdog hardware

Table 2-6. Power Switch Hardware Table

Table 2-8 through Table 2-10 show a variety of hardware components for an administrator to choose from. An individual cluster does *not* require all of the components listed in these tables.

Hardware	Quantity	Description	Required
Network interface	One for each network connection	Each network connection requires a network interface installed in a member.	Yes
Network switch or hub	One	A network switch or hub allows connection of multiple members to a network.	No
Network cable	One for each network interface	A conventional network cable, such as a cable with an RJ45 connector, connects each network interface to a network switch or a network hub.	Yes

Table 2-7. Network Hardware Table

Hardware	Quantity	Description	Required
Host bus adapter	One per member	To connect to shared disk storage, install either a parallel SCSI or a Fibre Channel host bus adapter in a PCI slot in each cluster member. For parallel SCSI, use a low voltage differential (LVD) host bus adapter. Adapters have either HD68 or VHDCI connectors. Host-bus adapter based RAID cards are only supported if they correctly support multi-host operation. At the time of publication, there were no fully tested host-bus adapter based RAID cards.	Yes
External disk storage enclosure	At least one	Use Fibre Channel or single-initiator parallel SCSI to connect the cluster members to a single or dual-controller RAID array. To use single-initiator buses, a RAID controller must have multiple host ports and provide simultaneous access to all the logical units on the host ports. To use a dual-controller RAID array, a logical unit must fail over from one controller to the other in a way that is transparent to the OS. SCSI RAID arrays that provide simultaneous access to all logical units on the host ports are recommended. To ensure symmetry of device IDs and LUNs, many RAID arrays with dual redundant controllers must be configured in an active/passive mode. Refer to Section 2.4.4 *Configuring Shared Disk Storage* for more information.	Yes

Hardware	Quantity	Description	Required
SCSI cable	One per member	SCSI cables with 68 pins connect each host bus adapter to a storage enclosure port. Cables have either HD68 or VHDCI connectors. Cables vary based on adapter type.	Only for parallel SCSI configurations
SCSI terminator	As required by hardware configuration	For a RAID storage enclosure that uses "out" ports (such as FlashDisk RAID Disk Array) and is connected to single-initiator SCSI buses, connect terminators to the "out" ports to terminate the buses.	Only for parallel SCSI configurations and only as necessary for termination
Fibre Channel hub or switch	One or two	A Fibre Channel hub or switch may be required.	Only for some Fibre Channel configurations
Fibre Channel cable	As required by hardware configuration	A Fibre Channel cable connects a host bus adapter to a storage enclosure port, a Fibre Channel hub, or a Fibre Channel switch. If a hub or switch is used, additional cables are needed to connect the hub or switch to the storage adapter ports.	Only for Fibre Channel configurations

Table 2-8. Shared Disk Storage Hardware Table

Hardware	Quantity	Description	Required
Network interface	Two for each member	Each Ethernet connection requires a network interface card installed on all cluster members.	No
Network crossover cable	One for each channel	A network crossover cable connects a network interface on one member to a network interface on other cluster members, creating an Ethernet connection for communicating heartbeat.	Only for a redundant Ethernet connection (use of channel-bonded Ethernet connection is preferred)

Table 2-9. Point-To-Point Ethernet Connection Hardware Table

Hardware	Quantity	Description	Required
UPS system	One or more	*Uninterruptible power supply* (UPS) systems protect against downtime if a power outage occurs. UPS systems are highly recommended for cluster operation. Connect the power cables for the shared storage enclosure and both power switches to redundant UPS systems. Note, a UPS system must be able to provide voltage for an adequate period of time, and should be connected to its own power circuit.	Strongly recommended for availability

Table 2-10. UPS System Hardware Table

Hardware	Quantity	Description	Required
Terminal server	One	A terminal server enables you to manage many members from one remote location.	No
KVM	One	A KVM enables multiple members to share one keyboard, monitor, and mouse. Cables for connecting members to the switch depend on the type of KVM.	No

Table 2-11. Console Switch Hardware Table

2.2. Setting Up the Members

After identifying the cluster hardware components described in Section 2.1 *Choosing a Hardware Configuration*, set up the basic cluster hardware and connect the members to the optional console switch and network switch or hub. Follow these steps:

1. In all members, install the required network adapters and host bus adapters. Refer to Section 2.2.1 *Installing the Basic Cluster Hardware* for more information about performing this task.
2. Set up the optional console switch and connect it to each member. Refer to Section 2.2.2 *Setting Up a Console Switch* for more information about performing this task.

   If a console switch is not used, then connect each member to a console terminal.
3. Set up the optional network switch or hub and use conventional network cables to connect it to the members and the terminal server (if applicable). Refer to Section 2.2.3 *Setting Up a Network Switch or Hub* for more information about performing this task.

   If a network switch or hub is not used, then conventional network cables should be used to connect each member and the terminal server (if applicable) to a network.

After performing the previous tasks, install Red Hat Enterprise Linux as described in Section 2.3 *Installing and Configuring Red Hat Enterprise Linux*.

2.2.1. Installing the Basic Cluster Hardware

Members must provide the CPU processing power and memory required by applications.

In addition, members must be able to accommodate the SCSI or Fibre Channel adapters, network interfaces, and serial ports that the hardware configuration requires. Systems have a limited number of pre-installed serial and network ports and PCI expansion slots. Table 2-12 helps determine how much capacity the member systems employed require.

Cluster Hardware Component	Serial Ports	Network Slots	PCI Slots
SCSI or Fibre Channel adapter to shared disk storage			One for each bus adapter

Cluster Hardware Component	Serial Ports	Network Slots	PCI Slots
Network connection for client access and Ethernet heartbeat pings		One for each network connection	
Point-to-point Ethernet connection for 2-node clusters (optional)		One for each connection	
Terminal server connection (optional)	One		

Table 2-12. Installing the Basic Cluster Hardware

Most systems come with at least one serial port. If a system has graphics display capability, it is possible to use the serial console port for a power switch connection. To expand your serial port capacity, use multi-port serial PCI cards. For multiple-member clusters, use a network power switch.

Also, ensure that local system disks are not on the same SCSI bus as the shared disks. For example, use two-channel SCSI adapters, such as the Adaptec 39160-series cards, and put the internal devices on one channel and the shared disks on the other channel. Using multiple SCSI cards is also possible.

Refer to the system documentation supplied by the vendor for detailed installation information. Refer to Appendix D *Supplementary Hardware Information* for hardware-specific information about using host bus adapters in a cluster.

Figure 2-2 shows the bulkhead of a sample member and the external cable connections for a typical cluster configuration.

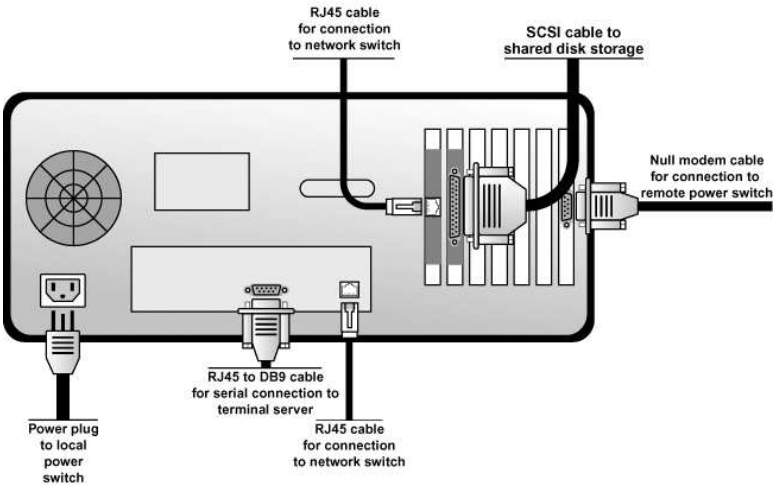


Figure 2-2. Typical External Cabling for a Cluster Member

2.2.2. Setting Up a Console Switch

Although a console switch is not required for cluster operation, it can be used to facilitate member management and eliminate the need for separate monitors, mouses, and keyboards for each cluster member. There are several types of console switches.

For example, a terminal server enables connection to serial consoles and management of many members from a remote location. For a low-cost alternative, use a KVM (keyboard, video, and mouse) switch, which enables multiple members to share one keyboard, monitor, and mouse. A KVM switch is suitable for configurations in which GUI access to perform system management tasks is preferred.

Set up the console switch according to the documentation provided by the vendor.

After the console switch has been set up, connect it to each cluster member. The cables used depend on the type of console switch. For example, a Cyclades terminal server uses RJ45 to DB9 crossover cables to connect a serial port on each member to the terminal server.

2.2.3. Setting Up a Network Switch or Hub

A network switch or hub, although not required for operating a two-node cluster, can be used to facilitate cluster and client system network operations. Clusters of more than two members require a switch or hub.

Set up a network switch or hub according to the documentation provided by the vendor.

After setting up the network switch or hub, connect it to each member by using conventional network cables. A terminal server, if used, is connected to the network switch or hub through a network cable.

2.3. Installing and Configuring Red Hat Enterprise Linux

After the setup of basic cluster hardware, proceed with installation of Red Hat Enterprise Linux on each member and ensure that all systems recognize the connected devices. Follow these steps:

1. Install Red Hat Enterprise Linux on all cluster members. Refer to *Red Hat Enterprise Linux Installation Guide* for instructions.

   In addition, when installing Red Hat Enterprise Linux, it is *strongly recommended* to do the following:

   - Gather the IP addresses for the members and for the bonded Ethernet ports, before installing Red Hat Enterprise Linux. Note that the IP addresses for the bonded Ethernet ports can be private IP addresses, (for example, 10.*x.x.x*).
   - Do not place local file systems (such as `/`, `/etc`, `/tmp`, and `/var`) on shared disks or on the same SCSI bus as shared disks. This helps prevent the other cluster members from accidentally mounting these file systems, and also reserves the limited number of SCSI identification numbers on a bus for cluster disks.
   - Place `/tmp` and `/var` on different file systems. This may improve member performance.
   - When a member boots, be sure that the member detects the disk devices in the same order in which they were detected during the Red Hat Enterprise Linux installation. If the devices are not detected in the same order, the member may not boot.
   - When using RAID storage configured with Logical Unit Numbers (LUNs) greater than zero, it is necessary to enable LUN support by adding the following to `/etc/modules.conf`:

     ```
     options scsi_mod max_scsi_luns=255
     ```

After modifying `modules.conf`, it is necessary to rebuild the initial ram disk using `mkinitrd`. Refer to the *Red Hat Enterprise Linux System Administration Guide* for more information about creating ramdisks using `mkinitrd`.

2. Reboot the members.
3. When using a terminal server, configure Red Hat Enterprise Linux to send console messages to the console port.
4. Edit the `/etc/hosts` file on each cluster member and include the IP addresses used in the cluster or ensure that the addresses are in DNS. Refer to Section 2.3.1 *Editing the `/etc/hosts` File* for more information about performing this task.
5. Decrease the alternate kernel boot timeout limit to reduce boot time for members. Refer to Section 2.3.2 *Decreasing the Kernel Boot Timeout Limit* for more information about performing this task.
6. Ensure that no login (or `getty`) programs are associated with the serial ports that are being used for the remote power switch connection (if applicable). To perform this task, edit the `/etc/inittab` file and use a hash symbol (#) to comment out the entries that correspond to the serial ports used for the remote power switch. Then, invoke the `init q` command.
7. Verify that all systems detect all the installed hardware:
   - Use the `dmesg` command to display the console startup messages. Refer to Section 2.3.3 *Displaying Console Startup Messages* for more information about performing this task.
   - Use the `cat /proc/devices` command to display the devices configured in the kernel. Refer to Section 2.3.4 *Displaying Devices Configured in the Kernel* for more information about performing this task.
8. Verify that the members can communicate over all the network interfaces by using the `ping` command to send test packets from one member to another.
9. If intending to configure Samba services, verify that the required RPM packages for Samba services are installed.

2.3.1. Editing the `/etc/hosts` File

The `/etc/hosts` file contains the IP address-to-hostname translation table. The `/etc/hosts` file on each member must contain entries for the following:

- IP addresses and associated hostnames for all cluster members
- IP addresses and associated hostnames for the point-to-point Ethernet heartbeat connections (these can be private IP addresses)

As an alternative to the `/etc/hosts` file, naming services such as DNS or NIS can be used to define the host names used by a cluster. However, to limit the number of dependencies and optimize availability, it is strongly recommended to use the `/etc/hosts` file to define IP addresses for cluster network interfaces.

The following is an example of an `/etc/hosts` file on a member:

```
127.0.0.1          localhost.localdomain   localhost
193.186.1.81       cluster2.example.com      cluster2
10.0.0.1           ecluster2.example.com     ecluster2
193.186.1.82       cluster3.example.com      cluster3
10.0.0.2           ecluster3.example.com     ecluster3
```

The previous example shows the IP addresses and hostnames for two members (*cluster2* and *cluster3*), and the private IP addresses and hostnames for the Ethernet interface (*ecluster2* and *ecluster3*) used for the point-to-point heartbeat connection on each member.

Verify correct formatting of the local host entry in the `/etc/hosts` file to ensure that it does not include non-local systems in the entry for the local host. An example of an incorrect local host entry that includes a non-local system (*server1*) is shown next:

```
127.0.0.1     localhost.localdomain     localhost server1
```

An Ethernet connection may not operate properly if the format of the `/etc/hosts` file is not correct. Check the `/etc/hosts` file and correct the file format by removing non-local systems from the local host entry, if necessary.

Note that each network adapter must be configured with the appropriate IP address and netmask.

The following example shows a portion of the output from the `/sbin/ifconfig` command on a cluster member:

```
eth0      Link encap:Ethernet  HWaddr 00:00:BC:11:76:93
          inet addr:192.186.1.81  Bcast:192.186.1.245  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:65508254 errors:225 dropped:0 overruns:2 frame:0
          TX packets:40364135 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:19 Base address:0xfce0

eth1      Link encap:Ethernet  HWaddr 00:00:BC:11:76:92
          inet addr:10.0.0.1  Bcast:10.0.0.245  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:18 Base address:0xfcc0
```

The previous example shows two network interfaces on a cluster member: *eth0* (the network interface for the member) and *eth1* (the network interface for the point-to-point Ethernet connection).

You may also add the IP addresses for the cluster members to your DNS server. Refer to the *Red Hat Enterprise Linux Reference Guide* for information on configuring DNS, or consult your network administrator.

2.3.2. Decreasing the Kernel Boot Timeout Limit

It is possible to reduce the boot time for a member by decreasing the kernel boot timeout limit. During the Red Hat Enterprise Linux boot sequence, the boot loader allows for specifying an alternate kernel to boot. The default timeout limit for specifying a kernel is ten seconds.

To modify the kernel boot timeout limit for a member, edit the appropriate files as follows:

When using the GRUB boot loader, the timeout parameter in `/boot/grub/grub.conf` should be modified to specify the appropriate number of seconds for the `timeout` parameter. To set this interval to 3 seconds, edit the parameter to the following:

```
timeout = 3
```

When using the LILO or ELILO boot loaders, edit the `/etc/lilo.conf` file (on x86 systems) or the `elilo.conf` file (on Itanium systems) and specify the desired value (in tenths of a second) for the `timeout` parameter. The following example sets the timeout limit to three seconds:

```
timeout = 30
```

To apply any changes made to the `/etc/lilo.conf` file, invoke the `/sbin/lilo` command.

On an Itanium system, to apply any changes made to the `/boot/efi/efi/redhat/elilo.conf` file, invoke the `/sbin/elilo` command.

2.3.3. Displaying Console Startup Messages

Use the `dmesg` command to display the console startup messages. Refer to the dmesg(8) man page for more information.

The following example of output from the `dmesg` command shows that two external SCSI buses and nine disks were detected on the member. (Lines with backslashes display as one line on most screens):

```
May 22 14:02:10 storage3 kernel: scsi0 : Adaptec AHA274x/284x/294x \
       (EISA/VLB/PCI-Fast SCSI) 5.1.28/3.2.4
May 22 14:02:10 storage3 kernel:
May 22 14:02:10 storage3 kernel: scsi1 : Adaptec AHA274x/284x/294x \
              (EISA/VLB/PCI-Fast SCSI) 5.1.28/3.2.4
May 22 14:02:10 storage3 kernel:
May 22 14:02:10 storage3 kernel: scsi : 2 hosts.
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST39236LW         Rev: 0004
May 22 14:02:11 storage3 kernel: Detected scsi disk sda at scsi0, channel 0, id 0, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdb at scsi1, channel 0, id 0, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdc at scsi1, channel 0, id 1, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdd at scsi1, channel 0, id 2, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sde at scsi1, channel 0, id 3, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdf at scsi1, channel 0, id 8, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdg at scsi1, channel 0, id 9, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdh at scsi1, channel 0, id 10, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: SEAGATE   Model: ST318203LC        Rev: 0001
May 22 14:02:11 storage3 kernel: Detected scsi disk sdi at scsi1, channel 0, id 11, lun 0
May 22 14:02:11 storage3 kernel:   Vendor: Dell      Model: 8 BAY U2W CU      Rev: 0205
May 22 14:02:11 storage3 kernel:   Type:   Processor \
                          ANSI SCSI revision: 03
May 22 14:02:11 storage3 kernel: scsi1 : channel 0 target 15 lun 1 request sense \
       failed, performing reset.
May 22 14:02:11 storage3 kernel: SCSI bus is being reset for host 1 channel 0.
May 22 14:02:11 storage3 kernel: scsi : detected 9 SCSI disks total.
```

The following example of the `dmesg` command output shows that a quad Ethernet card was detected on the member:

```
May 22 14:02:11 storage3 kernel: 3c59x.c:v0.99H 11/17/98 Donald Becker
May 22 14:02:11 storage3 kernel: tulip.c:v0.91g-ppc 7/16/99 becker@cesdis.gsfc.nasa.gov
May 22 14:02:11 storage3 kernel: eth0: Digital DS21140 Tulip rev 34 at 0x9800, \
       00:00:BC:11:76:93, IRQ 5.
May 22 14:02:12 storage3 kernel: eth1: Digital DS21140 Tulip rev 34 at 0x9400, \
       00:00:BC:11:76:92, IRQ 9.
May 22 14:02:12 storage3 kernel: eth2: Digital DS21140 Tulip rev 34 at 0x9000, \
       00:00:BC:11:76:91, IRQ 11.
May 22 14:02:12 storage3 kernel: eth3: Digital DS21140 Tulip rev 34 at 0x8800, \
```

```
00:00:BC:11:76:90, IRQ 10.
```

2.3.4. Displaying Devices Configured in the Kernel

To be sure that the installed devices, including serial and network interfaces, are configured in the kernel, use the `cat /proc/devices` command on each member. Use this command to also determine if there is raw device support installed on the member. For example:

```
Character devices:
  1 mem
  2 pty
  3 ttyp
  4 ttyS
  5 cua
  7 vcs
 10 misc
 19 ttyC
 20 cub
128 ptm
136 pts
162 raw

Block devices:
  2 fd
  3 ide0
  8 sd
 65 sd
```

The previous example shows:

- Onboard serial ports (`ttyS`)
- Serial expansion card (`ttyC`)
- Raw devices (`raw`)
- SCSI devices (`sd`)

2.4. Setting Up and Connecting the Cluster Hardware

After installing Red Hat Enterprise Linux, set up the cluster hardware components and verify the installation to ensure that the members recognize all the connected devices. Note that the exact steps for setting up the hardware depend on the type of configuration. Refer to Section 2.1 *Choosing a Hardware Configuration* for more information about cluster configurations.

To set up the cluster hardware, follow these steps:

1. Shut down the members and disconnect them from their power source.
2. Set up the bonded Ethernet channels, if applicable. Refer to Section 2.4.1 *Configuring Ethernet Channel Bonding* for more information.
3. When using power switches, set up the switches and connect each member to a power switch. Refer to Section 2.4.2 *Configuring Power Switches* for more information.

In addition, it is recommended to connect each power switch (or each member's power cord if not using power switches) to a different UPS system. Refer to Section 2.4.3 *Configuring UPS Systems* for information about using optional UPS systems.

4. Set up the shared disk storage according to the vendor instructions and connect the members to the external storage enclosure. Refer to Section 2.4.4 *Configuring Shared Disk Storage* for more information about performing this task.

   In addition, it is recommended to connect the storage enclosure to redundant UPS systems. Refer to Section 2.4.3 *Configuring UPS Systems* for more information about using optional UPS systems.

5. Turn on power to the hardware, and boot each cluster member. During the boot-up process, enter the BIOS utility to modify the member setup, as follows:

   - Ensure that the SCSI identification number used by the HBA is unique for the SCSI bus it is attached to. Refer to Section D.5 *SCSI Identification Numbers* for more information about performing this task.
   - Enable or disable the onboard termination for each host bus adapter, as required by the storage configuration. Refer to Section 2.4.4 *Configuring Shared Disk Storage* and Section D.3 *SCSI Bus Termination* for more information about performing this task.
   - Enable the member to automatically boot when it is powered on.

6. Exit from the BIOS utility, and continue to boot each member. Examine the startup messages to verify that the Red Hat Enterprise Linux kernel has been configured and can recognize the full set of shared disks. Use the `dmesg` command to display console startup messages. Refer to Section 2.3.3 *Displaying Console Startup Messages* for more information about using the `dmesg` command.
7. Verify that the members can communicate over each point-to-point Ethernet connection by using the `ping` command to send packets over each network interface.
8. Set up the shared cluster partitions on the shared disk storage. Refer to Section 2.4.4.3 *Configuring Shared Cluster Partitions* for more information about performing this task.

2.4.1. Configuring Ethernet Channel Bonding

Ethernet channel bonding in a no-single-point-of-failure cluster system allows for a fault tolerant network connection by combining two Ethernet devices into one virtual device. The resulting channel bonded interface ensures that in the event that one Ethernet device fails, the other device will become active. This type of channel bonding, called an *active-backup* policy allows connection of both bonded devices to one switch or can allow each Ethernet device to be connected to separate hubs or switches, which eliminates the single point of failure in the network hub/switch.

Channel bonding requires each cluster member to have two Ethernet devices installed. When it is loaded, the bonding module uses the MAC address of the first enslaved network device and assigns that MAC address to the other network device if the first device fails link detection.

To configure two network devices for channel bonding, perform the following:

1. Create a bonding devices in `/etc/modules.conf`. For example:

```
alias bond0 bonding
options bonding miimon=100 mode=1
```

   This loads the bonding device with the `bond0` interface name, as well as passes options to the bonding driver to configure it as an active-backup master device for the enslaved network interfaces.

2. Edit the `/etc/sysconfig/network-scripts/ifcfg-ethX` configuration file for both eth0 and eth1 so that the files show identical contents. For example:

```
DEVICE=ethX
USERCTL=no
ONBOOT=yes
MASTER=bond0
SLAVE=yes
BOOTPROTO=none
```

   This will enslave eth*X* (replace *X* with the assigned number of the Ethernet devices) to the bond0 master device.

3. Create a network script for the bonding device (for example, `/etc/sysconfig/network-scripts/ifcfg-bond0`), which would appear like the following example:

```
DEVICE=bond0
USERCTL=no
ONBOOT=yes
BROADCAST=192.168.1.255
NETWORK=192.168.1.0
NETMASK=255.255.255.0
GATEWAY=192.168.1.1
IPADDR=192.168.1.10
```

4. Reboot the system for the changes to take effect. Alternatively, manually load the bonding device and restart the network. For example:

```
/sbin/insmod /lib/modules/‘uname -r‘/kernel/drivers/net/bonding/bonding.o \
       miimon=100 mode=1
/sbin/service network restart
```

For more information about channel bonding, refer to the high-availability section of the *Linux Ethernet Bonding Driver Mini-Howto*, available in:

```
/usr/src/linux-2.4/Documentation/networking/bonding.txt
```



Note

You must have the `kernel-source` package installed in order to view the *Linux Ethernet Bonding Driver Mini-Howto*.

2.4.2. Configuring Power Switches

Power switches enable a member to power-cycle another member before restarting its services as part of the failover process. The ability to remotely disable a member ensures data integrity is maintained under any failure condition. It is recommended that production environments use power switches or watchdog timers in the cluster configuration. Only development (test) environments should use a configuration without power switches. Refer to Section 2.1.3 *Choosing the Type of Power Controller* for a description of the various types of power switches. Note that within this section, the general term "power switch" also includes watchdog timers.

In a cluster configuration that uses physical power switches, each member's power cable is connected to a power switch through either a serial or network connection (depending on switch type). When failover occurs, a member can use this connection to power-cycle another member before restarting its services.

Power switches protect against data corruption if an unresponsive (or hanging) member becomes responsive after its services have failed over, and issues I/O to a disk that is also receiving I/O from another member. In addition, if a quorum daemon fails on a member, the member is no longer able to monitor the shared cluster partitions. If power switches or watchdog timers are not used in the cluster, then this error condition may result in services being run on more than one member, which can cause data corruption and possibly system crashes.

It is strongly recommended to use power switches in a cluster. However, administrators who are aware of the risks may choose to set up a cluster without power switches.

A member may hang for a few seconds if it is swapping or has a high system workload. For this reason, adequate time is allowed prior to concluding another member has failed (typically 15 seconds).

If a member determines that a hung member is down, and power switches are used in the cluster, that member power-cycles the hung member before restarting its services. Clusters configured to use watchdog timers self-reboot under most system hangs. This causes the hung member to reboot in a clean state and prevent it from issuing I/O and corrupting service data.

Hung members reboot themselves either due to a watchdog firing, failure to send heartbeat packets, or — in the case a member has no physical power switch — loss of quorum status.

Hung members may be rebooted by other members if they are attached to a power switch. If the hung member never becomes responsive and no power switches are in use, then a manual reboot is required.

When used, power switches must be set up according to the vendor instructions. However, some cluster-specific tasks may be required to use a power switch in the cluster. Refer to Section D.1 *Setting Up Power Controllers* for detailed information on power switches (including information about watchdog timers). Be sure to take note of any caveats or functional attributes of specific power switches. Note that the cluster-specific information provided in this manual supersedes the vendor information.

When cabling power switches, take special care to ensure that each cable is plugged into the appropriate outlet. This is crucial because there is no independent means for the software to verify correct cabling. Failure to cable correctly can lead to an incorrect member being power cycled, or for one member to inappropriately conclude that it has successfully power cycled another cluster member.

After setting up the power switches, perform these tasks to connect them to the members:

1. Connect the power cable for each member to a power switch.
2. Connect each member to the power switch. The cable used for the connection depends on the type of power switch. Serial-attached power switches use null modem cables, while a network-attached power switches require an Ethernet patch cable.
3. Connect the power cable for each power switch to a power source. It is recommended to connect each power switch to a different UPS system. Refer to Section 2.4.3 *Configuring UPS Systems* for more information.

After the installation of the cluster software, test the power switches to ensure that each member can power-cycle the other member before starting the cluster. Refer to Section 3.11.2 *Testing the Power Switches* for information.

2.4.3. Configuring UPS Systems

Uninterruptible power supplies (UPS) provide a highly-available source of power. Ideally, a redundant solution should be used that incorporates multiple UPS systems (one per server). For maximal fault-tolerance, it is possible to incorporate two UPS systems per server as well as APC Automatic Transfer Switches to manage the power and shutdown management of the server. Both solutions are solely dependent on the level of availability desired.

It is not recommended to use a single UPS infrastructure as the sole source of power for the cluster. A UPS solution dedicated to the cluster is more flexible in terms of manageability and availability.

A complete UPS system must be able to provide adequate voltage and current for a prolonged period of time. While there is no single UPS to fit every power requirement, a solution can be tailored to fit a particular configuration.

If the cluster disk storage subsystem has two power supplies with separate power cords, set up two UPS systems, and connect one power switch (or one member's power cord if not using power switches) and one of the storage subsystem's power cords to each UPS system. A redundant UPS system configuration is shown in Figure 2-3.

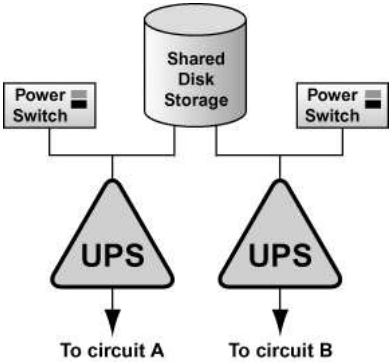


Figure 2-3. Redundant UPS System Configuration

An alternative redundant power configuration is to connect the power switches (or the members' power cords) and the disk storage subsystem to the same UPS system. This is the most cost-effective configuration, and provides some protection against power failure. However, if a power outage occurs, the single UPS system becomes a possible single point of failure. In addition, one UPS system may not be able to provide enough power to all the attached devices for an adequate amount of time. A single UPS system configuration is shown in Figure 2-4.

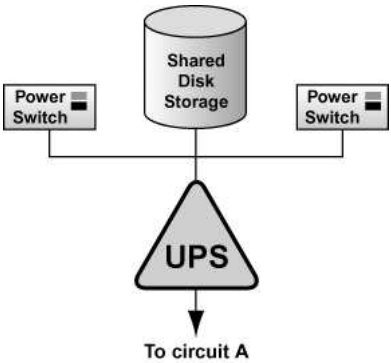


Figure 2-4. Single UPS System Configuration

Many vendor-supplied UPS systems include Red Hat Enterprise Linux applications that monitor the operational status of the UPS system through a serial port connection. If the battery power is low, the monitoring software initiates a clean system shutdown. As this occurs, the cluster software is properly stopped, because it is controlled by a SysV runlevel script (for example, `/etc/rc.d/init.d/clumanager`).

Refer to the UPS documentation supplied by the vendor for detailed installation information.

2.4.4. Configuring Shared Disk Storage

In a cluster, shared disk storage is used to hold service data and two partitions (primary and shadow) that store cluster state information. Because this storage must be available to all members, it cannot be located on disks that depend on the availability of any one member. Refer to the vendor documentation for detailed product and installation information.

There are some factors to consider when setting up shared disk storage in a cluster:

- External RAID

  It is strongly recommended to use use RAID 1 (mirroring) to make service data and the shared cluster partitions highly available. Optionally, parity RAID can also be employed for high-availability. Do not use RAID 0 (striping) alone for shared partitions because this reduces storage availability.
- Multi-initiator SCSI configurations

  Multi-initiator SCSI configurations are not supported due to the difficulty in obtaining proper bus termination.
- The Red Hat Enterprise Linux device name for each shared storage device must be the same on each member. For example, a device named `/dev/sdc` on one member must be named `/dev/sdc` on the other cluster members. Using identical hardware for all members usually ensures that these devices are named the same.
- A disk partition can be used by only one cluster service.
- Do not include any file systems used in a cluster service in the member's local `/etc/fstab` files, because the cluster software must control the mounting and unmounting of service file systems.
- For optimal performance of shared file systems, make sure to specify a 4 KB block size with the `-b` option to `mke2fs`. A smaller block size can cause long `fsck` times. Refer to Section 2.4.4.6 *Creating File Systems*.

The following list details *parallel SCSI* requirements, and must be adhered to when parallel SCSI buses are employed in a cluster environment:

- SCSI buses must be terminated at each end, and must adhere to length and hot plugging restrictions.
- Devices (disks, host bus adapters, and RAID controllers) on a SCSI bus must have a unique SCSI identification number.

Refer to Section D.2 *SCSI Bus Configuration Requirements* for more information.

It is *strongly recommended* to connect the storage enclosure to redundant UPS systems for a highly-available source of power. Refer to Section 2.4.3 *Configuring UPS Systems* for more information.

Refer to Section 2.4.4.1 *Setting Up a Single-initiator SCSI Bus* and Section 2.4.4.2 *Setting Up a Fibre Channel Interconnect* for more information about configuring shared storage.

After setting up the shared disk storage hardware, partition the disks and then either create file systems or raw devices on the partitions. Two raw devices must be created for the primary and the shadow shared cluster partitions. Refer to Section 2.4.4.3 *Configuring Shared Cluster Partitions*,

Section 2.4.4.4 *Partitioning Disks*, Section 2.4.4.5 *Creating Raw Devices*, and Section 2.4.4.6 *Creating File Systems* for more information.

2.4.4.1. Setting Up a Single-initiator SCSI Bus

A single-initiator SCSI bus has only one member connected to it, and provides host isolation and better performance than a multi-initiator bus. Single-initiator buses ensure that each member is protected from disruptions due to the workload, initialization, or repair of the other members.

When using a single- or dual-controller RAID array that has multiple host ports and provides simultaneous access to all the shared logical units from the host ports on the storage enclosure, the setup of the single-initiator SCSI buses to connect each cluster member to the RAID array is possible. If a logical unit can fail over from one controller to the other, the process must be transparent to the operating system. Note that some RAID controllers restrict a set of disks to a specific controller or port. In this case, single-initiator bus setups are not possible.

A single-initiator bus must adhere to the requirements described in Section D.2 *SCSI Bus Configuration Requirements*.

To set up a single-initiator SCSI bus configuration, the following is required:

- Enable the on-board termination for each host bus adapter.
- Enable the termination for each RAID controller.
- Use the appropriate SCSI cable to connect each host bus adapter to the storage enclosure.

Setting host bus adapter termination is usually done in the adapter BIOS utility during member boot. To set RAID controller termination, refer to the vendor documentation. Figure 2-5 shows a configuration that uses two single-initiator SCSI buses.

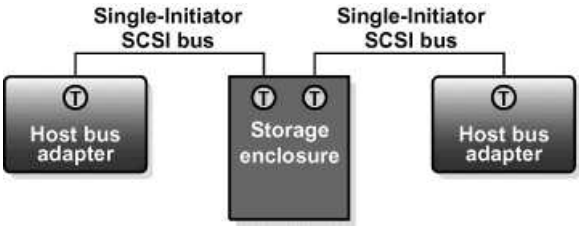


Figure 2-5. Single-initiator SCSI Bus Configuration

Figure 2-6 shows the termination in a single-controller RAID array connected to two single-initiator SCSI buses.

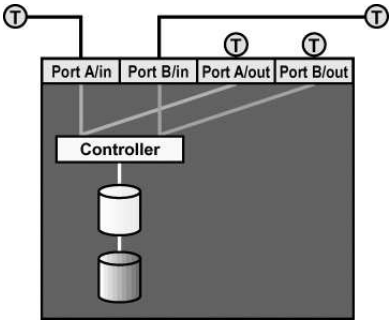


Figure 2-6. Single-controller RAID Array Connected to Single-initiator SCSI Buses

Figure 2-7 shows the termination in a dual-controller RAID array connected to two single-initiator SCSI buses.

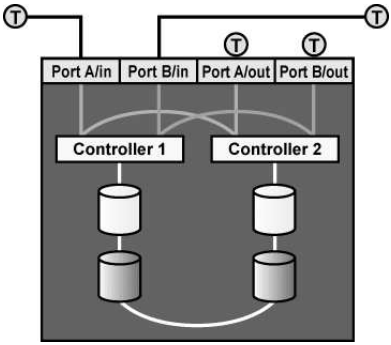


Figure 2-7. Dual-controller RAID Array Connected to Single-initiator SCSI Buses

2.4.4.2. Setting Up a Fibre Channel Interconnect

Fibre Channel can be used in either single-initiator or multi-initiator configurations.

A single-initiator Fibre Channel interconnect has only one member connected to it. This may provide better host isolation and better performance than a multi-initiator bus. Single-initiator interconnects ensure that each member is protected from disruptions due to the workload, initialization, or repair of the other member.

If employing a RAID array that has multiple host ports, and the RAID array provides simultaneous access to all the shared logical units from the host ports on the storage enclosure, set up single-initiator Fibre Channel interconnects to connect each member to the RAID array. If a logical unit can fail over from one controller to the other, the process must be transparent to the operating system.

Figure 2-8 shows a single-controller RAID array with two host ports and the host bus adapters connected directly to the RAID controller, without using Fibre Channel hubs or switches. When using this type of single-initiator Fibre Channel connection, your RAID controller must have a separate host port for each cluster member.

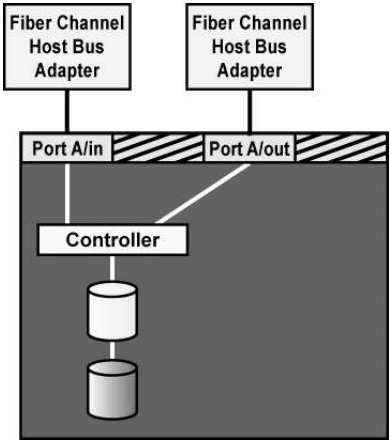


Figure 2-8. Single-controller RAID Array Connected to Single-initiator Fibre Channel Interconnects

The external RAID array must have a separate SCSI channel for each cluster member. In clusters with more than two members, connect each member to a different SCSI channel on the RAID array, using a single-initiator SCSI bus as shown in Figure 2-8.

To connect multiple cluster members to the same host port on the RAID array, use an FC hub or switch. In this case, each HBA is connected to the hub or switch, and the hub or switch is connected to a host port on the RAID controller.

A Fibre Channel hub or switch is also required with a dual-controller RAID array with two host ports on each controller. This configuration is shown in Figure 2-9. Additional cluster members may be connected to either Fibre Channel hub or switch shown in the diagram. Some RAID arrays include a built-in hub so that each host port is already connected to each of the internal RAID controllers. In this case, an additional external hub or switch may not be needed.

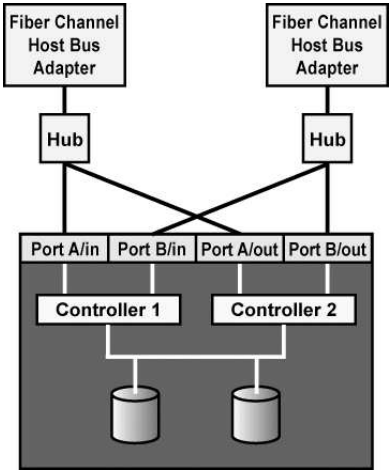


Figure 2-9. Dual-controller RAID Array Connected to Single-initiator Fibre Channel Interconnects

2.4.4.3. Configuring Shared Cluster Partitions

Two raw devices on shared disk storage must be created for the primary shared partition and the shadow shared partition. Each shared partition must have a minimum size of 10 MB. The amount of data in a shared partition is constant; it does not increase or decrease over time.

The shared partitions are used to hold cluster state information, including the following:

- Cluster lock states
- Service states
- Configuration information

Periodically, each member writes the state of its services to shared storage. In addition, the shared partitions contain a version of the cluster configuration file. This ensures that each member has a common view of the cluster configuration.

If the primary shared partition is corrupted, the cluster members read the information from the shadow (or backup) shared partition and simultaneously repair the primary partition. Data consistency is maintained through checksums, and any inconsistencies between the partitions are automatically corrected.

If a member is unable to write to both shared partitions at startup time, it is not allowed to join the cluster. In addition, if an active member can no longer write to both shared partitions, the member removes itself from the cluster by rebooting (and may be remotely power cycled by a healthy member).

The following are *shared partition requirements*:

- Both partitions must have a minimum size of 10 MB.
- Shared partitions must be raw devices. They cannot contain file systems.
- Shared partitions can be used only for cluster state and configuration information.

The following are *recommended guidelines* for configuring the shared partitions:

- It is strongly recommended to set up a RAID subsystem for shared storage, and use RAID 1 (mirroring) to make the logical unit that contains the shared partitions highly available. Optionally, parity RAID can be used for high availability. Do not use RAID 0 (striping) alone for shared partitions.
- Place both shared partitions on the same RAID set, or on the same disk if RAID is not employed, because both shared partitions must be available for the cluster to run.
- Do not put the shared partitions on a disk that contains heavily-accessed service data. If possible, locate the shared partitions on disks that contain service data that is rarely accessed.

Refer to Section 2.4.4.4 *Partitioning Disks* and Section 2.4.4.5 *Creating Raw Devices* for more information about setting up the shared partitions.

Refer to Section 3.5 *Editing the `rawdevices` File* for information about editing the `rawdevices` file to bind the raw character devices to the block devices each time the members boot.

2.4.4.4. Partitioning Disks

After shared disk storage hardware has been set up, partition the disks so they can be used in the cluster. Then, create file systems or raw devices on the partitions. For example, two raw devices must be created for the shared partitions using the guidelines described in Section 2.4.4.3 *Configuring Shared Cluster Partitions*.

Use `parted` to modify a disk partition table and divide the disk into partitions. While in `parted`, use the `p` to display the partition table and the `mkpart` command to create new partitions. The following example shows how to use `parted` to create a partition on disk:

- Invoke `parted` from the shell using the command `parted` and specifying an available shared disk device. At the `(parted)` prompt, use the `p` to display the current partition table. The output should be similar to the following:

```
Disk geometry for /dev/sda: 0.000-4340.294 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
```

- Decide on how large a partition is required. Create a partition of this size using the `mkpart` command in `parted`. Although the `mkpart` does not create a file system, it normally requires a file system type at partition creation time. `parted` uses a range on the disk to determine partition size; the size is the space between the end and the beginning of the given range. The following example shows how to create two partitions of 20 MB each on an empty disk.

```
(parted) mkpart primary ext3 0 20
(parted) mkpart primary ext3 20 40
(parted) p
Disk geometry for /dev/sda: 0.000-4340.294 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.030     21.342  primary
2         21.343     38.417  primary
```

- When more than four partitions are required on a single disk, it is necessary to create an *extended partition*. If an extended partition is required, the `mkpart` also performs this task. In this case, it is not necessary to specify a file system type.



Note

Only one extended partition may be created, and the extended partition *must* be one of the four primary partitions.

```
(parted) mkpart extended 40 2000
(parted) p
```

```
Disk geometry for /dev/sda: 0.000-4340.294 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.030     21.342  primary
2         21.343     38.417  primary
3         38.417   2001.952  extended
```

- An extended partition allows the creation of *logical partitions*inside of it. The following example shows the division of the extended partition into two logical partitions.

```
(parted) mkpart logical ext3 40 1000
(parted) p
Disk geometry for /dev/sda: 0.000-4340.294 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.030     21.342  primary
2         21.343     38.417  primary
3         38.417   2001.952  extended
5         38.447    998.841  logical
(parted) mkpart logical ext3 1000 2000
(parted) p
Disk geometry for /dev/sda: 0.000-4340.294 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
1          0.030     21.342  primary
2         21.343     38.417  primary
3         38.417   2001.952  extended
5         38.447    998.841  logical
6        998.872   2001.952  logical
```

- A partition may be removed using `parted`'s `rm` command. For example:

```
(parted) rm 1
(parted) p
Disk geometry for /dev/sda: 0.000-4340.294 megabytes
Disk label type: msdos
Minor    Start       End     Type      Filesystem  Flags
2         21.343     38.417  primary
3         38.417   2001.952  extended
5         38.447    998.841  logical
6        998.872   2001.952  logical
```

- After all required partitions have been created, exit `parted` using the `quit` command. If a partition was added, removed, or changed while both members are powered on and connected to the shared storage, reboot the other member for it to recognize the modifications. After partitioning a disk, format the partition for use in the cluster. For example, create the file systems or raw devices for shared partitions. Refer to Section 2.4.4.5 *Creating Raw Devices* and Section 2.4.4.6 *Creating File Systems* for more information.

  For basic information on partitioning hard disks at installation time, refer to the *Red Hat Enterprise Linux Installation Guide*.

2.4.4.5. Creating Raw Devices

After partitioning the shared storage disks, create raw devices on the partitions. File systems are block devices (for example, `/dev/sda1`) that cache recently-used data in memory to improve performance. Raw devices do not utilize system memory for caching. Refer to Section 2.4.4.6 *Creating File Systems* for more information.

Red Hat Enterprise Linux supports raw character devices that are not hard-coded against specific block devices. Instead, Red Hat Enterprise Linux uses a character major number (currently 162) to implement a series of unbound raw devices in the `/dev/raw/` directory. Any block device can have a character raw device front-end, even if the block device is loaded later at run time.

To create a raw device, edit the /etc/sysconfig/rawdevices file to bind a raw character device to the appropriate block device to enable the raw device to be opened, read, and written.

Shared partitions and some database applications require raw devices, because these applications perform their own buffer caching for performance purposes. Shared partitions cannot contain file systems because if state data was cached in system memory, the members would not have a consistent view of the state data.

Raw character devices must be bound to block devices each time a member boots. To ensure that this occurs, edit the /etc/sysconfig/rawdevices file and specify the shared partition bindings. If using a raw device in a cluster service, use this file to bind the devices at boot time. Refer to Section 3.5 *Editing the rawdevices File* for more information.

After editing /etc/sysconfig/rawdevices, the changes take effect either by rebooting or by execute the following command:

```
service rawdevices restart
```

Query all the raw devices by using the command raw -aq. The output should be similar to the following:

```
/dev/raw/raw1   bound to major 8, minor 17
/dev/raw/raw2   bound to major 8, minor 18
```

Note that, for raw devices, no cache coherency exists between the raw device and the block device. In addition, requests must be 512-byte aligned both in memory and on disk. For example, the standard dd command cannot be used with raw devices because the memory buffer that the command passes to the write system call is not aligned on a 512-byte boundary.

For more information on using the raw command, refer to the raw(8) man page.



Note

The same raw device names (for example, /dev/raw/raw1 and /dev/raw/raw2) must be used on all cluster members.

2.4.4.6. Creating File Systems

Use the mkfs command to create an ext3 file system. For example:

```
mke2fs -j -b 4096 /dev/sde3
```

For optimal performance of shared file systems, make sure to specify a 4 KB block size with the -b option to mke2fs. A smaller block size can cause long fsck times.

Chapter 3.

Cluster Configuration

After installing and configuring the cluster hardware, the cluster system software and cluster configuration software can be installed.

3.1. Installing the Red Hat Cluster Suite Packages

The `clumanager` and `redhat-config-cluster` packages are required to configure the Red Hat Cluster Manager. Perform the following instructions to install the Red Hat Cluster Suite on your Red Hat Enterprise Linux system.

3.1.1. Installation with the Package Management Tool

1. Insert the Red Hat Cluster Suite CD in your CD-ROM drive. If you are using a graphical desktop, the CD will automatically run the **Package Management Tool**. Click **Forward** to continue.



Figure 3-1. Package Management Tool

2. Check the box for the Red Hat Cluster Suite, and click the **Details** link to the package descriptions.

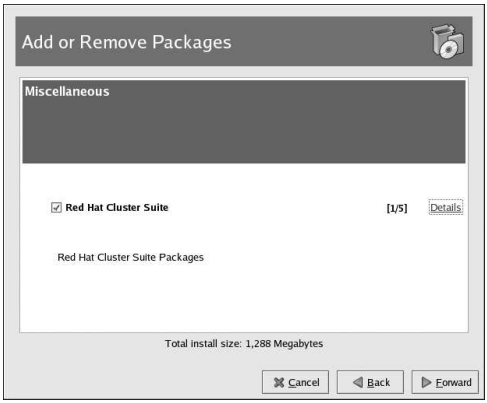


Figure 3-2. Choosing Package Group

3. While viewing the package group details, check the box next to the packages to install. Click **Close** when finished.

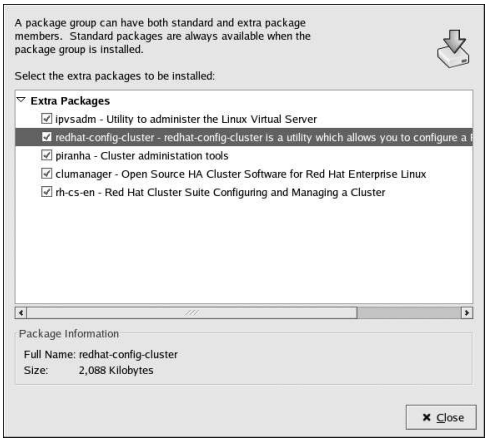


Figure 3-3. Choosing Packages for Installation

4. The **Package Management Tool** shows an overview of the packages to be installed. Click **Forward** to install the packages.
5. When the installation is complete, click **Finish** to exit the **Package Management Tool**.

3.1.2. Installation with `rpm`

If you are not using a graphical desktop environment, you can install the packages manually using the `rpm` utility at a shell prompt.

Insert the Red Hat Cluster Suite CD into the CD-ROM drive. Log into a shell prompt, change to the `RedHat/RPMS/` directory on the CD, and type the following commands as root (replacing *<version>* and *<arch>* with the version and architecture of the packages to install):

```
rpm --Uvh clumanager-<version>.<arch>.rpm

rpm --Uvh redhat-config-cluster-<version>.noarch.rpm
```

3.2. Installation Notes for Red Hat Enterprise Linux 2.1 Users



Warning

Make sure the cluster service is *not* running on any members before performing the following procedure.

If you are currently running cluster services on Red Hat Enterprise Linux 2.1 and you want to install Red Hat Enterprise Linux 3 on your system while preserving your cluster configuration, perform the following steps on the Red Hat Enterprise Linux 2.1 system to backup the cluster configuration before installing Red Hat Enterprise Linux 3:

1. Stop the cluster service by invoking the command `service cluster stop` on all cluster members.
2. Remove the cluster database file (`/etc/cluster.conf`) from all members, saving a copy with a different name (for example, `/etc/cluster.conf.sav`) to a separate running system or portable storage such as a diskette.
3. Reinstall the Red Hat Enterprise Linux 2.1 system with Red Hat Enterprise Linux 3 on all members.
4. Install the Red Hat Cluster Suite using the method described in Section 3.1 *Installing the Red Hat Cluster Suite Packages*.
5. On all members, install the applications to be managed as services by the cluster software.
6. Restore the cluster database file you saved in step 2 to one member, naming the file `/etc/cluster.conf`.
7. Invoke the `/usr/sbin/cluster-convert` command to convert the `/etc/cluster.conf` to `/etc/cluster.xml`.
8. Configure shared storage on the cluster member by running `/usr/sbin/shutil -i` which initializes the shared storage. Then run `/usr/sbin/shutil -s /etc/cluster.xml` to read the shared storage information from the file and write it to the shared state.
9. Start the cluster service on this member using the command `/sbin/service clumanager start`.
10. Copy the `/etc/cluster.xml` file to the other member using the `scp` command.
11. Start the cluster service on the other members.

3.3. The Cluster Configuration Tool

Red Hat Cluster Manager consists of the following RPM packages:

- `clumanager` — This package consists of the software that is responsible for cluster operation (including the cluster daemons).
- `redhat-config-cluster` — This package contains the **Cluster Configuration Tool** and the **Cluster Status Tool**, which allow for the configuration of the cluster and the display of the current status of the cluster and its members and services.

You can use either of the following methods to access the **Cluster Configuration Tool**:

- Select **Main Menu => System Settings => Server Settings => Cluster**.
- At a shell prompt, type the `redhat-config-cluster` command.

The first time that the application is started, the **Cluster Configuration Tool** is displayed. After you complete the cluster configuration, the command starts the **Cluster Status Tool** by default. To access the **Cluster Configuration Tool** from the **Cluster Status Tool**, select **Cluster => Configure**.

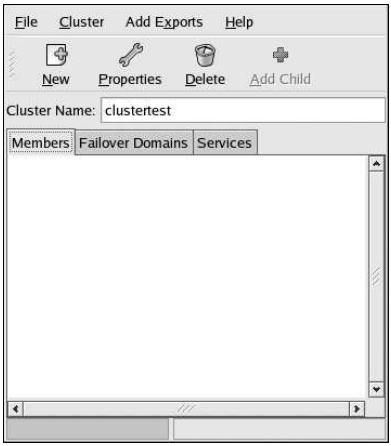


Figure 3-4. The Cluster Configuration Tool

The following tabbed sections are available within the **Cluster Configuration Tool**:

- **Members** — Use this section to add members to the cluster and optionally configure a power controller connection for any given member.
- **Failover Domains** — Use this section to establish one or more subsets of the cluster members for specifying which members are eligible to run a service in the event of a system failure. (Note that the use of failover domains is optional.)
- **Services** — Use this section to configure one or more services to be managed by the cluster. As you specify an application service, the relationship between the service and its IP address, device special file, mount point, and NFS exports is represented by a hierarchical structure. The parent-child relationships in the **Cluster Configuration Tool** reflect the organization of the service information in the `/etc/cluster.xml` file.

Warning

Do not manually edit the contents of the `/etc/cluster.xml` file.

Do not simultaneously run the **Cluster Configuration Tool** on multiple members. (It is permissible to run the **Cluster Status Tool** on more than one member at a time.)

The **Cluster Configuration Tool** stores information about the cluster service and daemons, cluster members, and cluster services in the `/etc/cluster.xml` configuration file. The cluster configuration file is created the first time the **Cluster Configuration Tool** is started.

Save the configuration at any point (using **File => Save**) while running the **Cluster Configuration Tool**. When **File => Quit** is selected, it prompts you to save changes if any unsaved changes to the configuration are detected.

Note

When you save the cluster configuration for the first time using the **Cluster Configuration Tool** and exit the application, the next time you the cluster (either by choosing **Main Menu** => **System Settings** => **Server Settings** => **Cluster** or by running `redhat-config-cluster` from a shell prompt) the **Cluster Status Tool** will display by default. The **Cluster Status Tool** displays the status of the cluster service, cluster members, and application services, and shows statistics concerning service operation. If you need to further configure the cluster system, choose **Cluster** => **Configure** from the **Cluster Status Tool** menu.

The **Cluster Configuration Tool**) is used to configure cluster members, services, and cluster daemons. The **Cluster Status Tool** is used to monitor, start and stop the cluster members and services on particular members, and move an application service to another member.

The **Cluster Configuration Tool** uses a hierarchical tree structure to show relationships between components in the cluster configuration. A triangular icon to the left of a component name indicates that the component has children. To expand or collapse the portion of the tree below a component, click the triangle icon.

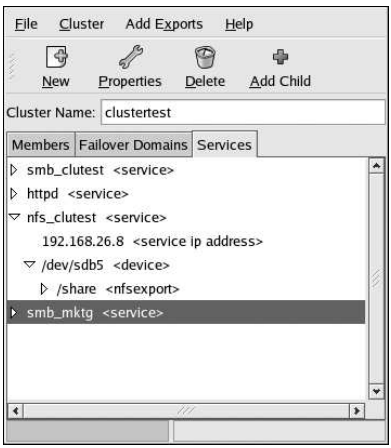


Figure 3-5. Cluster Configuration Tool

To display the properties of a component, select the component and click **Properties**. As a shortcut, you can display the properties of a component by double-clicking on the component name.

To view or modify the properties of the cluster daemons, choose **Cluster => Daemon Properties**.

3.4. Configuring the Cluster Software

Before describing the steps to configure the cluster software, the hierarchical structure of cluster members and services needs to be considered. Cluster members and services can be thought of in terms of a parent/child tree structure, as shown in Figure 3-6.

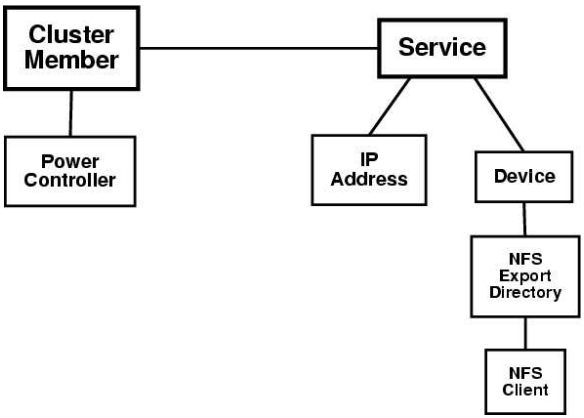


Figure 3-6. Cluster Configuration Structure

Each member can have power controller children. Each service can have IP address children. Each Service can also have device children. Each device can have NFS export directory children. Each NFS export directory can have client children. This structure is reflected in the `/etc/cluster.xml` file syntax.

The steps to configure the cluster software consist of the following:

1. Edit the `/etc/sysconfig/rawdevices` file on all cluster members and specify the raw device special files and character devices for the primary and backup shared partitions as described in Section 2.4.4.3 *Configuring Shared Cluster Partitions* and Section 3.5 *Editing the `rawdevices` File*.
2. Run the **Cluster Configuration Tool** on one cluster member.
3. Enter a name for the cluster in the **Cluster Name** field. The name should be descriptive enough to distinguish it from other clusters and systems on your network (for example, **`nfs_cluster`** or **`httpd_cluster`**).
4. Choose **Cluster => Shared State**, and confirm that the **Raw Primary** and **Raw Shadow** settings match the settings specified in step 1. The default for **Raw Primary** is **`/dev/raw/raw1`**. The default for **Raw Shadow** is **`/dev/raw/raw2`**.
5. Choose **Cluster => Daemon Properties** to edit the daemon properties. Each daemon has its own tab. Update the daemon options to suit your preferences and operating environment, and click **OK** when done. Refer to Section 3.6 *Configuring Cluster Daemons* for more details.

6. Add member systems to the cluster by selecting the **Members** tab and clicking the **New** button. Refer to Section 3.7 *Adding and Deleting Members* for details.

   If the hardware configuration includes power controllers (power switches), then for each member connected to a power controller, configure that member's connection to the power controller by selecting the member and clicking **Add Child**. Refer to Section 3.8 *Configuring a Power Controller Connection* for more information.

7. Set up one or more failover domains, if needed, to restrict the members on which a service can run or restrict the order of members followed when a service fails over from one failover domain member to another (or both). Refer to Section 3.9 *Configuring a Failover Domain* for details.

8. Configure the application services to be managed by the cluster, specifying IP addresses, failover domain (if applicable) and user script that manages the service. Refer to Section 3.10 *Adding a Service to the Cluster* for more information.

9. Save cluster configuration changes by selecting **File => Save**. When you save the cluster configuration, the command `service clumanager reload` command is executed to cause the cluster software to load the changed configuration file.

10. Quit the application by selecting **File => Quit**.

    Running the **Cluster Configuration Tool** for the first time causes the cluster configuration file `/etc/cluster.xml` to be created automatically. When quiting after running it for the first time, you are prompted to **Press 'OK' to initialize Shared Storage**, which uses the shared partitions to pass quorum and service state information between cluster members. Click **OK** to initialize the shared storage.



Note

Shared storage *must* be initialized before starting the cluster service for the first time, or the cluster will not run properly.

11. After completing the cluster software configuration on one member, copy the configuration file `/etc/cluster.xml` to all other cluster members. The `scp` can be used to copy files from one host to another.

3.5. Editing the `rawdevices` File

The `/etc/sysconfig/rawdevices` file is used to map the raw devices for the shared partitions each time a cluster member boots. As part of the cluster software installation procedure, edit the `rawdevices` file on each member and specify the raw character devices and block devices for the primary and backup shared partitions. This must be done prior to running the **Cluster Configuration Tool**.

If raw devices are employed in a cluster service, the `rawdevices` file is also used to bind the devices at boot time. Edit the file and specify the raw character devices and block devices that you want to bind each time the member boots. To make the changes to the rawdevices file take effect without requiring a reboot, perform the following command:

```
/sbin/service rawdevices restart
```

The following is an example `rawdevices` file that designates two shared partitions:

```
# raw device bindings
# format:  <rawdev> <major> <minor>
#          <rawdev> <blockdev>
# example: /dev/raw/raw1 /dev/sda1
```

```
#          /dev/raw/raw2 8 5
/dev/raw/raw1 /dev/hda5
/dev/raw/raw2 /dev/hda6
```

Refer to Section 2.4.4.3 *Configuring Shared Cluster Partitions* for more information about setting up the shared partitions. Refer to Section 2.4.4.5 *Creating Raw Devices* for more information on using the raw command to bind raw character devices to block devices.



Note

The `rawdevices` configuration must be performed on all cluster members, and all members must use the same raw devices (from the previous example, `/dev/raw/raw1` and `/dev/raw/raw2`).

To check raw device configuration on the current cluster member, choose **Cluster => Shared State** in the **Cluster Configuration Tool**. The **Shared State** dialog is displayed, as shown in Figure 3-7.

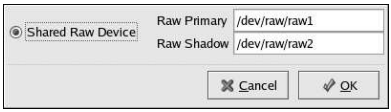


Figure 3-7. Verifying Shared State Configuration

3.6. Configuring Cluster Daemons

The Red Hat Cluster Manager provides the following daemons to monitor cluster operation:

- `cluquorumd` — Quorum daemon
- `clusvcmgrd` — Service manager daemon
- `clurmtabd` — Synchronizes NFS mount entries in `/var/lib/nfs/rmtab` with a private copy on a service's mount point
- `clulockd` — Global lock manager (the only client of this daemon is `clusvcmgrd`)
- `clumembd` — Membership daemon

Each of these daemons can be individually configured using the **Cluster Configuration Tool**. To access the **Cluster Daemon Properties** dialog box, choose **Cluster => Daemon Properties**.

The following sections explain how to configure cluster daemon properties. However, note that the default values are applicable to most configurations and do not need to be changed.

3.6.1. Configuring clumembd

On each cluster system, the `clumembd` daemon issues heartbeats (pings) across the point-to-point Ethernet lines to which the cluster members are connected.

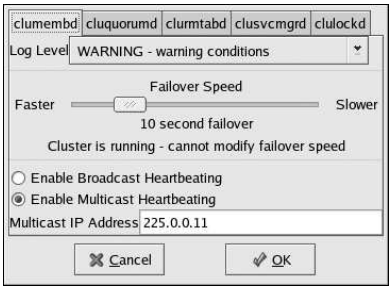


Figure 3-8. Configuring `clumembd`



Note

You can enable both broadcast heartbeating and multicast heartbeating, but *at least one* of these features must be used.

Multicast heartbeating over a channel-bonded Ethernet interface provides good fault tolerance and is recommended for availability.

You can specify the following properties for the `clumembd` daemon:

- **Log Level** — Determines the level of event messages that get logged to the cluster log file (by default `/var/log/messages`). Choose the appropriate logging level from the menu. Refer to Section 3.12 *Configuring `syslogd` Event Logging* for more information.
- **Failover Speed**—Determines the number of seconds that the cluster service waits before shutting down a non-responding member (that is, a member from which no heartbeat is detected). To set the failover speed, drag the slider bar. The default failover speed is 10 seconds.

  Note

  Setting a faster failover speed increases the likelihood of false shutdowns.

- Heartbeating — **Enable Broadcast Heartbeating** or **Enable Multicast Heartbeating** by clicking the corresponding radio button. Broadcast heartbeating specifies that the broadcast IP address is to be used by the `clumembd` daemon when emitting heartbeats.

  By default, the `clumembd` is configured to emit heartbeats via multicast. Multicast uses the network interface associated with the member's hostname for transmission of heartbeats.
- **Multicast IP Address** — Specifies the IP address to be used by the `clumembd` daemon over the multicast channel. This field is not editable if **Enable Broadcast Heartbeating** is checked. The default multicast IP address used by the cluster is 225.0.0.11.

3.6.2. Configuring cluquorumd

In a two-member cluster without a specified tiebreaker IP address, the `cluquorumd` daemon periodically writes a time-stamp and system status to a specific area on the primary and shadow shared

partitions. The daemon also reads the other member's timestamp and system status information from the primary shared partition or, if the primary partition is corrupted, from the shadow partition.

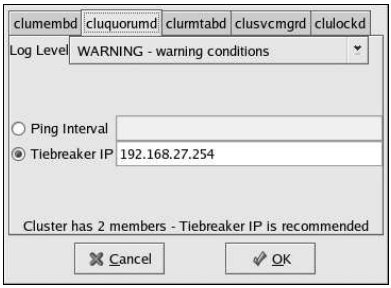


Figure 3-9. Configuring `cluquorumd`

You can specify the following properties for the `cluquorumd` daemon:

- **Log Level** — Determines the level of event messages that get logged to the cluster log file (by default `/var/log/messages`). Choose the appropriate logging level from the menu. Refer to Section 3.12 *Configuring `syslogd` Event Logging* for more information.
- **Ping Interval** or **Tiebreaker IP** — **Ping Interval** is used for a disk-based heartbeat; specifies the number of seconds between the quorum daemon's updates to its on-disk status.

  Tiebreaker IP is a network-based heartbeat used to determine quorum, which is the ability to run services. The tiebreaker IP address is only checked when the cluster has an even split in a two- or four-member cluster. This IP address should be associated with a router that, during normal operation, can be reached by all members over the Ethernet interface used by the cluster software.

3.6.3. Configuring clurmtabd

The `clurmtabd` daemon synchronizes NFS mount entries in `/var/lib/nfs/rmtab` with a private copy on a service's mount point. The `clurmtabd` daemon runs only when a service with NFS exports is running.

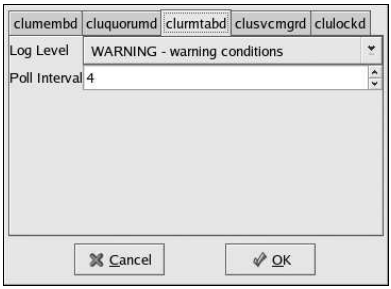


Figure 3-10. Configuring `clurmtabd`

You can specify the following properties for the `clurmtabd` daemon:

- **Log Level** — Determines the level of event messages that get logged to the cluster log file (by default, `/var/log/messages`). Choose the appropriate logging level from the menu. See Section 3.12 *Configuring `syslogd` Event Logging* for more information.
- **Poll Interval**—Specifies the number of seconds between polling cycles to synchronize the local NFS `rmtab` to the cluster `rmtab` on shared storage.

3.6.4. Configuring the clusvcmgrd daemon

On each cluster system, the `clusvcmgrd` service manager daemon responds to changes in cluster membership by stopping and starting services. You might notice, at times, that more than one `clusvcmgrd` process is running; separate processes are spawned for `start`, `stop`, and `monitoring` operations.

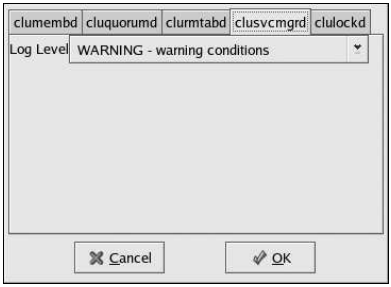


Figure 3-11. Configuring `clusvcmgrd`

You can specify the following properties for the `clusvcmgrd` daemon:

- **Log Level** — Determines the level of event messages that get logged to the cluster log file (by default, `/var/log/messages`). Choose the appropriate logging level from the menu. See Section 3.12 *Configuring `syslogd` Event Logging* for more information.

3.6.5. Configuring clulockd

The `clulockd` daemon manages the locks on files being accessed by cluster members.

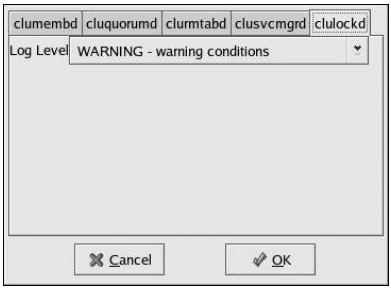


Figure 3-12. Configuring `clulockd`

You can specify the following properties for the `clulockd` daemon:

- **Log Level** — Determines the level of event messages that get logged to the cluster log file (by default, `/var/log/messages`). Choose the appropriate logging level from the menu. See Section 3.12 *Configuring `syslogd` Event Logging* for more information.

3.7. Adding and Deleting Members

The procedure to add a member to a cluster varies slightly, depending on whether the cluster is already running or is a newly-configured cluster.

3.7.1. Adding a Member to a New Cluster

To add a member to a new cluster, follow these steps:



Figure 3-13. Adding a Member to a New Cluster

1. Ensure that the **Members** tab is selected and click **New**. It prompts for a member name.
2. Enter the name or address of a system on the cluster subnet. Note that each member must be on the same subnet as the system on which you are running the **Cluster Configuration Tool** and must be defined either in DNS or in each cluster system's `/etc/hosts` file

   The system on which you are running the **Cluster Configuration Tool** must be explicitly added as a cluster member; the system is not automatically added to the cluster configuration as a result of running the **Cluster Configuration Tool**.
3. Leave **Enable SW Watchdog** checked. (A software watchdog timer enables a member to reboot itself.)
4. Click **OK**.

5. Choose **File => Save** to save the changes to the cluster configuration.

3.7.2. Adding a Member to a Running Cluster

To add a member to an existing cluster that is currently in operation, follow these steps:

1. Ensure that the cluster service is *not* running on the new member by invoking the `/sbin/service clumanager status` command. Invoke the `/sbin/service clumanager stop` command to stop the cluster service.
2. Ensure that the cluster service is running on all active cluster members. Run `/sbin/service clumanager start` to start the service on the existing cluster members.
3. On one of the running members, invoke the **Cluster Configuration Tool** to add the new member. Ensure that the **Members** tab is selected and click **New**.
4. It prompts for a member name. Enter the name of the new member. Note that each member must be on the same subnet as the system on which you are running the **Cluster Configuration Tool** and must be defined in DNS or in each cluster member's `/etc/hosts` file..
5. Leave **Enable SW Watchdog** checked. (A software watchdog timer enables a member to reboot itself.)
6. Click **OK**.
7. Choose **File => Save** to save the changes to the cluster configuration.
8. Copy the updated `/etc/cluster.xml` file (containing the newly-added member) to the new member.
9. Invoke the `service clumanager start` command to start the cluster service on the new member.

3.7.3. Deleting a Member from a Running Cluster

To delete a member from an existing cluster that is currently in operation, follow these steps:

1. Ensure that the cluster service is *not* running on the member to be deleted. Invoke the `service clumanager stop` command to stop the cluster service.
2. On one of the running members, invoke the **Cluster Configuration Tool** to delete the new member, as follows:
   a. Remove the member from any failover domains of which it is a member (see Section 3.9 *Configuring a Failover Domain*).
   b. Select the **Members** tab.
   c. Select the member to be deleted and click **Delete**.
   d. Click **Yes** to confirm deletion.

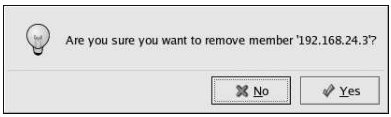


Figure 3-14. Deleting a Member

3. Choose **File => Save** to save the changes to the cluster configuration.

3.8. Configuring a Power Controller Connection

To ensure data integrity, only one member can run a service and access service data at one time. The use of power controllers (also called power switches) in the cluster hardware configuration enables a member to power-cycle another member before restarting that member's services during the failover process. This prevents more than one system from simultaneously accessing the same data and corrupting it. Although not required, it is recommended that you use power controllers to guarantee data integrity under all failure conditions. Watchdog timers are an optional variety of power control to ensure correct operation of service failover.

If the hardware configuration for the cluster includes one or more power controllers, configure the connection to a power controller for any given member as follows:

1. Select the member for which you want to configure a power controller connection and click **Add Child**. The **Power Controller** dialog box is displayed as shown in Figure 3-15.

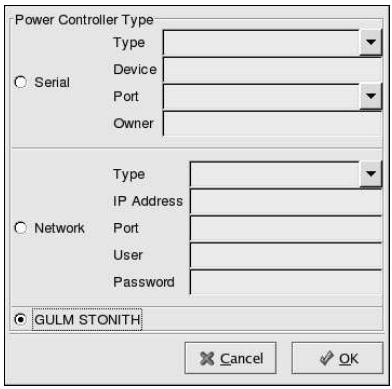


Figure 3-15. Configuring Power Controllers

2. Specify the following information depending on whether the controller is connected to the member through a serial port or through the network. Note that serial connections to power controllers are valid only in a two-member cluster.

Field	Description
Type	Type of serial power controller (only the `rps10` is supported in Red Hat Enterprise Linux 3).
Device	Physical device on the power controller's parent member that the power controller is plugged into; must be one of `/dev/ttyS0` through `/dev/ttyS9`.
Port	Port number on the power controller itself that is attached to the specified device.

Field	Description
Owner	The member that controls, or owns, the device. The owner is the member that can power off this device. Automatically defaults to the name of the other member. This field cannot be edited.

Table 3-1. Configuring a Serial Power Controller

Field	Description
Type	Type of network power controller; choose from the menu containing all supported types.
IP Address	IP of the network power controller itself.
Port	Specific port on the power controller that this member is attached to.
User	Username to be used when logging in to the power controller.
Password	Password to be used when logging in to the power controller.

Table 3-2. Configuring a Network Power Controller

Users of Red Hat GFS can choose the *Grand Unified Lock Manager* (GULM) driver as their power controller by clicking the **GULM STONITH** radio button. The GULM driver is used in conjunction with the Red Hat Cluster Manager failover subsystem and features performance gains in the detection and fencing initialization of hung or failed cluster members.

For more information about Red Hat GFS refer to the *Red Hat GFS Administrator's Guide*.

3. Click **OK**.
4. Choose **File => Save** to save the changes to the cluster configuration.

3.9. Configuring a Failover Domain

A failover domain is a named subset of cluster members that are eligible to run a service in the event of a system failure. A failover domain can have the following characteristics:

- Unrestricted — Allows you to specify that a subset of members are preferred, but that a service assigned to this domain can run on any available member.
- Restricted — Allows you to restrict the members that can run a particular service. If none of the members in a restricted failover domain are available, the service cannot be started (either manually or by the cluster software).
- Unordered — When a service is assigned to an unordered failover domain, the member on which the service runs is chosen from the available failover domain members with no priority ordering.
- Ordered — Allows you to specify a preference order among the members of a failover domain. The member at the top of the list is the most preferred, followed by the second member in the list, and so on.

By default, failover domains are unrestricted and unordered.

In a cluster with several members, using a restricted failover domain can minimize the work to set up the cluster to run a service (such as `httpd`, which requires you to set up the configuration identically on all members that run the service). Instead of setting up the entire cluster to run the service, you must set up only the members in the restricted failover domain that you associate with the service.



Tip

To implement the concept of a preferred member, create an unrestricted failover domain comprised of only one cluster member. By doing this, a service runs on the preferred member; in the event of a failure, the service fails over to any of the other members.

To add a failover domain to the cluster software configuration, follow these steps:

1. Select the **Failover Domains** tab and click **New**. The **Failover Domain** dialog box is displayed as shown in Figure 3-16.

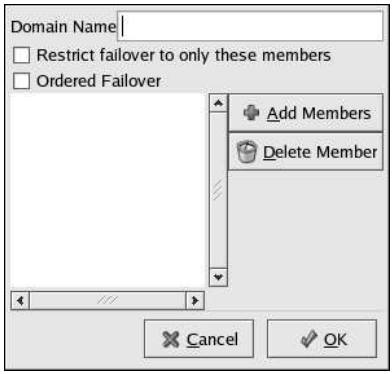


Figure 3-16. Configuring a Failover Domain

2. Enter a name for the domain in the **Domain Name** field. The name should be descriptive enough to distinguish its purpose relative to other names used on your network.
3. Check **Restrict failover to only these members** to prevent any member other than those listed from taking over a service assigned to this domain.
4. Check **Ordered Failover** if you want members to assume control of a failed service in a particular sequence; preference is indicated by the member's position in the list of members in the domain, with the most preferred member at the top.
5. Click **Add Members** to select the members for this failover domain. The **Failover Domain Member** dialog box is displayed.



Figure 3-17. Failover Domain Member

You can choose multiple members from the list by pressing either the [Shift] key while clicking the start and end of a range of members, or pressing the [Ctrl] key while clicking on noncontiguous members.

6. When finished selecting members from the list, click **OK**. The selected members are displayed on the **Failover Domain** list.
7. When **Ordered Failover** is checked, you can rearrange the order of the members in the domain by dragging the member name in the list box to the desired position. A thin, black line is displayed to indicate the new row position (when you release the mouse button).
8. When finished, click **OK**.
9. Choose **File => Save** to save the changes to the cluster configuration.

To remove a member from a failover domain, follow these steps:

1. On the **Failover Domains** tab, double-click the name of the domain you want to modify (or select the domain and click **Properties**).
2. In the **Failover Domain** dialog box, click the name of the member you want to remove from the domain and click **Delete Member**. (Members must be deleted one at a time.) You are prompted to confirm the deletion.
3. When finished, click **OK**.
4. Choose **File => Save** to save the changes to the cluster configuration.

3.10. Adding a Service to the Cluster

To add a service to the cluster, follow these steps:

1. Select the **Services** tab and click **New**. The **Service** dialog is displayed as shown in Figure 3-18.

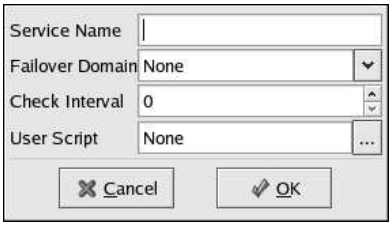


Figure 3-18. Adding a Service

2. Give the service a descriptive **Service Name** to distinguish its functionality relative to other services that may run on the cluster.
3. If you want to restrict the members on which this service is able to run, choose a failover domain from the **Failover Domain** list. (Refer to Section 3.9 *Configuring a Failover Domain* for instructions on how to configure a failover domain.)
4. Adjust the quantity in the **Check Interval** field, which sets the interval (in seconds) that the cluster infrastructure checks the status of a service. This field is only applicable if the service script is written to check the status of a service.
5. Specify a **User Script** that contains settings for starting, stopping, and checking the status of a service.
6. Specify service properties, including an available *floating IP address* (an address that can be transferred transparently from a failed member to a running member in the event of failover) and devices (which are configured as children of the service). For instructions, refer to Section 3.10.1 *Adding a Service IP Address* and Section 3.10.2 *Adding a Service Device*.
7. When finished, click **OK**.
8. Choose **File => Save** to save the changes to the cluster configuration.

The following sections describe service configuration in more detail.

3.10.1. Adding a Service IP Address

To specify a service IP address, follow these steps:

1. On the **Services** tab of the **Cluster Configuration Tool**, select the service you want to configure and click **Add Child**.
2. Select **Add Service IP Address** and click **OK**.
3. Specify an IP address (which must be resolvable by DNS but cannot be the IP address of a running service).
4. Optionally specify a netmask and broadcast IP address.
5. Optionally, check the **Monitor Link** box to enable or disable link status monitoring of the IP address resource (refer to Figure 3-19).

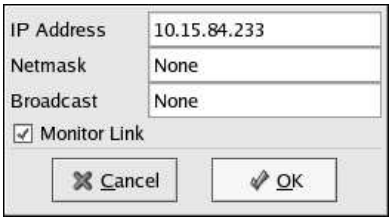


Figure 3-19. Optional Monitor Link

6. Choose **File => Save** to save the change to the `/etc/cluster.xml` configuration file.

The Monitor Link feature monitors the link status of the Ethernet device to which the relevant IP address is assigned and returns a status failure if the MII does not report that a link is detected. If you enable the Monitor Link feature, consider the following characteristics of the feature:

- Monitor links only corresponds to links on devices where cluster service IP addresses reside. All other system links are ignored.
- The Monitor Link feature is a per-service-IP-address attribute, and only monitors the link for the given service IP address.
- Monitor Link checks on start and every time a service check (`status`) run occurs. Service checks only occur if you have defined a service check interval. If your service check interval is 0, no link status monitoring occurs except at service start. (For more information about specifying check interval, refer to Section 4.1.1 *Gathering Service Information*.)
- If multiple service IP addresses are assigned to the same network device, the link is checked for each service IP assigned to that network device which has the Monitor Link option set.

3.10.2. Adding a Service Device

To specify a device for a service, follow these steps:

1. On the **Services** tab of the **Cluster Configuration Tool**, select the service you want to configure and click **Add Child**.
2. Select **Add Device** and click **OK**.
3. Specify a **Device Special File** (for example, `/dev/hda7`) and a mount point (for example, `/mnt/share`). Each device must have a unique device special file and a unique mount point within the cluster.
4. Specify a **Samba Share Name** for the device if it is intended to be a Samba export directory. If a **Samba Share Name** has been entered, when the user selects **File => Save**, a `/etc/samba/smb.conf.`*`sharename`* file is created (where *`sharename`* is the name of the Samba share), and will be used by Samba when the cluster starts the service. For each Samba share you create, an `/etc/samba/smb.conf.`*`sharename`* is created. Copy all of these files to the other cluster members before initializing the cluster service on those members. For more information about configuring a Samba share, refer to Section 6.6 *Setting Up a Samba Service*.
5. Specify a directory from which to mount the device in the **Mount Point** field. This directory should *not* be listed in `/etc/fstab` as it is automatically mounted by the Red Hat Cluster Manager when the service is started.
6. Choose a file system type from the **FS Type** list.

7. Optionally specify **Options** for the device. If you leave the **Options** field blank, the default mount options (`rw,suid,dev,exec,auto,nouser,async`) are used. Refer to the `mount` man page for a complete description of the options available for mount.
8. Check **Force Unmount** to force any application that has the specified file system mounted to be killed prior to disabling or relocating the service (when the application is running on the same member that is running the disabled or relocated service).
9. When finished, click **OK**.
10. Choose **File => Save** to save the change to the `/etc/cluster.xml` configuration file.

3.11. Checking the Cluster Configuration

To ensure that the cluster software has been correctly configured, use the following tools located in the `/usr/sbin` directory:

- Test the shared partitions and ensure that they are accessible.

  Invoke the `/usr/sbin/shutil` utility with the `-v` option to test the accessibility of the shared partitions. See Section 3.11.1 *Testing the Shared Partitions* for more information.
- Test the operation of the power switches.

  If power switches are used in the cluster hardware configuration, run the `clufence` command on each member to ensure that it can remotely power-cycle the other member. Do not run this command while the cluster software is running. See Section 3.11.2 *Testing the Power Switches* for more information.
- Ensure that all members are running the same software version.

  Invoke the `rpm -q clumanager` command and `rpm -q redhat-config-cluster` command on each member to display the revision of the installed cluster software RPMs.

The following section explains the cluster utilities in further detail.

3.11.1. Testing the Shared Partitions

The shared partitions must refer to the same physical device on all members. Invoke the `/usr/sbin/shutil` utility with the `-v` command to test the shared partitions and verify that they are accessible.

If the command succeeds, run the `/usr/sbin/shutil -p /cluster/header` command on all members to display a summary of the header data structure for the shared partitions. If the output is different on the members, the shared partitions do not point to the same devices on all members. Check to make sure that the raw devices exist and are correctly specified in the `/etc/sysconfig/rawdevices` file. See Section 2.4.4.3 *Configuring Shared Cluster Partitions* for more information.

The following example shows that the shared partitions refer to the same physical device on cluster members `clu1.example.com` and `clu2.example.com` via the `/usr/sbin/shutil -p /cluster/header` command:

```
/cluster/header is 140 bytes long
SharedStateHeader {
        ss_magic = 0x39119fcd
        ss_timestamp = 0x000000003ecbc215 (14:14:45 May 21 2003)
        ss_updateHost = clu1.example.com
```

All fields in the output from the `/usr/sbin/shutil -p /cluster/header` command should be the same when run on all cluster members. If the output is *not* the same on all members, perform the following:

- Examine the `/etc/sysconfig/rawdevices` file on each member and ensure that the raw character devices and block devices for the primary and backup shared partitions have been accurately specified. If they are not the same, edit the file and correct any mistakes. Then re-run the **Cluster Configuration Tool**. See Section 3.5 *Editing the `rawdevices` File* for more information.
- Ensure that you have created the raw devices for the shared partitions on each member. See Section 2.4.4.3 *Configuring Shared Cluster Partitions* for more information.
- To determine the bus configuration on each member, examine the system startup messages by running `dmesg |less` to the point where the system probes the SCSI subsystem. Verify that all members identify the same shared storage devices and assign them the same name.
- Verify that a member is not attempting to mount a file system on the shared partition. For example, make sure that the actual device (for example, `/dev/sdb1`) is not included in an `/etc/fstab` file.

After performing these tasks, re-run the `/usr/sbin/shutil` utility with the `-p` option.

3.11.2. Testing the Power Switches

If either network-attached or serial-attached power switches are employed in the cluster hardware configuration, install the cluster software and invoke the `clufence` command to test the power switches. Invoke the command on each member to ensure that it can remotely power-cycle the other member. If testing is successful, then the cluster can be started.

The `clufence` command can accurately test a power switch only if the cluster software is not running. This is due to the fact that for serial attached switches, only one program at a time can access the serial port that connects a power switch to a member. When the `clufence` command is invoked, it checks the status of the cluster software. If the cluster software is running, the command exits with a message to stop the cluster software.

The `clufence` command line options are as follows:

- **`-d`** — Turn on debugging
- **`-f`** — Fence (power off) member
- **`-u`** — Unfence (power on) member
- **`-r`** — Reboot (power cycle) member
- **`-s`** — Check status of all switches controlling member

When testing power switches, the first step is to ensure that each cluster member can successfully communicate with its attached power switch. The following output of the `clufence` command shows that the cluster member is able to communicate with its power switch:

```
[27734] info: STONITH: rps10 at /dev/ttyS0, port 0 controls clumember1.example.com
[27734] info: STONITH: rps10 at /dev/ttyS0, port 1 controls clumember2.example.com
```

In the event of an error in the `clufence` output, check the following:

- For serial attached power switches:
  - Verify that the device special file for the remote power switch connection serial port (for example, `/dev/ttyS0`) is specified correctly in the cluster configuration file; in the **Cluster Configuration Tool**, display the **Power Controller** dialog box to check the serial port value. If necessary,

use a terminal emulation package such as **minicom** to test if the cluster member can access the serial port.

- Ensure that a non-cluster program (for example, a `getty` program) is not using the serial port for the remote power switch connection. You can use the `lsof` command to perform this task.
- Check that the cable connection to the remote power switch is correct. Verify that the correct type of cable is used (for example, an RPS-10 power switch requires a null modem cable), and that all connections are securely fastened.
- Verify that any physical dip switches or rotary switches on the power switch are set properly.

- For network based power switches:
  - Verify that the network connection to network-based power switches is operational. Most switches have a link light that indicates connectivity.
  - It should be possible to `ping` the network power switch; if not, then the switch may not be properly configured for its network parameters.
  - Verify that the correct password and login name (depending on switch type) have been specified in the cluster configuration file (as established by running the **Cluster Configuration Tool** and viewing the properties specified in the **Power Controller** dialog box). A useful diagnostic approach is to verify Telnet access to the network switch using the same parameters as specified in the cluster configuration.

After successfully verifying communication with the switch, attempt to power cycle the other cluster member. Prior to doing this, we recommend you verify that the other cluster member is not actively performing any important functions (such as serving cluster services to active clients). Running the command `clufence -f clumember2.example.com` displays the following output upon a successful shutdown and *fencing* operation (which means that the system does not receive power from the power switch until the system has been unfenced):

```
[7397] info: STONITH: rps10 at /dev/ttyS0, port 0 controls clumember1.example.com
[7397] info: STONITH: rps10 at /dev/ttyS0, port 1 controls clumember2.example.com
[7397] notice: STONITH: clumember2.example.com has been fenced!
```

3.11.3. Displaying the Cluster Software Version

Ensure that all members in the cluster are running the same version of the Red Hat Cluster Manager software.

To display the version of the **Cluster Configuration Tool** and the **Cluster Status Tool**, use either of the following methods:

- Choose **Help => About**. The **About** dialog includes the version numbers.
- Invoke the following commands:
  ```
  rpm -q redhat-config-cluster
  rpm -q clumanager
  ```
- The version of the `clumanager` package can also be determined by invoking the `clustat -v` command.

3.12. Configuring `syslogd` Event Logging

It is possible to edit the `/etc/syslog.conf` file to enable the cluster to log events to a file that is different from the `/var/log/messages` log file. Logging cluster messages to a separate file helps to diagnose problems more clearly.

The members use the `syslogd` daemon to log cluster-related events to a file, as specified in the `/etc/syslog.conf` file. The log file facilitates diagnosis of problems in the cluster. It is recommended to set up event logging so that the `syslogd` daemon logs cluster messages only from the member on which it is running. Therefore, you need to examine the log files on all members to get a comprehensive view of the cluster.

The `syslogd` daemon logs messages from the cluster daemons, which all default to severity level 4 (**warning**). Refer to Section 3.6 *Configuring Cluster Daemons* for more information on cluster daemons.

The importance of an event determines the severity level of the log entry. Important events should be investigated before they affect cluster availability. The cluster can log messages with the following severity levels, listed in order of severity level:

- **EMERG** —The member is unusable (emergency).
- **ALERT** —Action must be taken immediately to address the problem.
- **CRIT** —A critical condition has occurred.
- **ERR** —An error has occurred.
- **WARN** —A significant event that may require attention has occurred.
- **NOTICE** —A significant, but normal, event has occurred.
- **INFO** —An insignificant, but normal, cluster operation has occurred.
- **DEBUG** —Diagnostic output detailing cluster operations (typically not of interest to administrators.)

Examples of log file entries are as follows:

```
Jul 18 20:24:39 clu1 clufence[7397]: <info> STONITH: rps10 at /dev/ttyS0,\
   port 0 controls clu1
Jul 18 20:24:39 clu1 clufence[7397]: <info> STONITH: rps10 at /dev/ttyS0,\
   port 1 controls clu2
Jul 18 20:24:53 clu1 clufence[7397]: Port 0 being turned off.
Jul 18 20:24:53 clu1 clufence[7397]: <notice> STONITH: clu2 has been fenced!
Jul 18 20:51:03 clu1 clufence[30780]: <info> STONITH: rps10 at/dev/ttyS0,\
   port 0 controls clu1
Jul 18 20:51:03 clu1 clufence[30780]: <info> STONITH: rps10 at /dev/ttyS0,\
   port 1 controls clu2
Jul 18 20:51:17 clu1 clufence[30780]: Port 0 being turned on.
Jul 18 20:51:17 clu1 clufence[30780]: <notice> STONITH: clu2 is no longer fenced off.
     [1]         [2]      [3]               [4]         [5]
```

Each entry in the log file contains the following information:

- [1] Date and time
- [2] Hostname
- [3] Cluster resource or daemon
- [4] Severity
- [5] Message

After configuring the cluster software, optionally edit the `/etc/syslog.conf` file to enable the cluster to log events to a file that is different from the default log file, `/var/log/messages`. The cluster utilities and daemons log their messages using a syslog tag called *`local4`*. Using a cluster-specific log file facilitates cluster monitoring and problem solving.

To prevent cluster events from being logged to the `/var/log/messages` file, add `local4.none` to the following line in the `/etc/syslog.conf` file:

```
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;local4.none   /var/log/messages
```

To direct the cluster logging facility to log cluster events in the `/var/log/cluster` file, add lines similar to the following to the `/etc/syslog.conf` file:

```
#
# Cluster messages coming in on local4 go to /var/log/cluster
#
local4.*                        /var/log/cluster
```

To apply the previous changes, restart `syslogd` with the `service syslog restart` command.

In addition, it is possible to modify the severity level of the events that are logged by the individual cluster daemons; refer to Section 3.6 *Configuring Cluster Daemons* and the man page for `syslog.conf` for more information.

To rotate the cluster log file according to the frequency specified in the `/etc/logrotate.conf` file (the default is weekly), add `/var/log/cluster` to the first line of the `/etc/logrotate.d/syslog` file. For example:

```
/var/log/messages /var/log/secure /var/log/maillog /var/log/spooler
/var/log/boot.log /var/log/cron /var/log/cluster {
   sharedscripts
   postrotate
       /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2>
       /dev/null || true
   endscript
}
```

Chapter 4.

Service Administration

The following sections describe how to display, enable, disable, modify, relocate, and delete a service, and how to handle services that fail to start. Examples of setting up specific types of services are provided.

4.1. Configuring a Service

After setting up disk storage and installing applications to be managed by the cluster, you can configure services to manage these applications and resources by using the **Cluster Configuration Tool**.

To configure a service, follow these steps:

1. If applicable, create a script to start, stop, and check the status of the application used in the service. Refer to Section 4.1.2 *Creating Service Scripts* for information.
2. Gather information about service resources and properties. Refer to Section 4.1.1 *Gathering Service Information* for information.
3. Set up the file systems or raw devices that the service uses. Refer to Section 4.1.3 *Configuring Service Disk Storage* for information.
4. Ensure that the application software can run on each member (in either the associated failover domain, if used, or in the cluster) and that the service script, if any, can start and stop the service application. Refer to Section 4.1.4 *Verifying Application Software and Service Scripts* for upgrade instructions.
5. If upgrading from an existing cluster, back up the `/etc/cluster.conf` file. Refer to Section 3.2 *Installation Notes for Red Hat Enterprise Linux 2.1 Users* for upgrade instructions.
6. Start the **Cluster Configuration Tool** and add services, specifying the information about the service resources and properties obtained in step 2.
7. Save your configuration. Saving the settings on one cluster member propogates to other cluster members automatically.

For more information about adding a cluster service, refer to the following:

- Section 5.1 *Setting Up an Oracle Service*
- Section 5.3 *Setting Up a MySQL Service*
- Section 6.1 *Setting Up an NFS Service*
- Section 6.6 *Setting Up a Samba Service*
- Chapter 7 *Setting Up Apache HTTP Server*

4.1.1. Gathering Service Information

Before configuring a service, gather all available information about the service resources and properties. In some cases, it is possible to specify multiple resources for a service (for example, multiple IP addresses and disk devices).

The service properties and resources that you can specify using the **Cluster Configuration Tool** are described in Table 4-1.

Property	Description
Service name	Each service must have a unique name. A service name can consist of one to 63 characters and must consist of a combination of letters (either uppercase or lowercase), integers, underscores, periods, and dashes (hyphens). A service name *must* begin with a letter or an underscore.
Failover domain	Identify the members on which to run the service by associating the service with an existing failover domain. When ordered failover is enabled, if the member on which the service is running fails, the service is automatically relocated to the next member on the ordered member list. (Order of preference is established by the sequence of member names in the failover domain list). Refer to Section 3.9 *Configuring a Failover Domain* for more information.
Check interval	Specifies the frequency (in seconds) that the member checks the health of the application associated with the service. For example, when you specify a nonzero check interval for an NFS or Samba service, Red Hat Cluster Manager verifies that the necessary NFS or Samba daemons are running. For other types of services, Red Hat Cluster Manager checks the return status after calling the `status` clause of the application service script. By default, check interval is 0, indicating that service monitoring is disabled.
User script	If applicable, specify the full path name for the script that is used to start and stop the service. Refer to Section 4.1.2 *Creating Service Scripts* for more information.
IP address	One or more Internet protocol (IP) addresses may be assigned to a service. This IP address (sometimes called a "floating" IP address) is different from the IP address associated with the host name Ethernet interface for a member, because it is automatically relocated along with the service resources when failover occurs. If clients use this IP address to access the service, they do not know which member is running the service, and failover is transparent to the clients. Note that cluster members must have network interface cards configured in the IP subnet of each IP address used in a service. Netmask and broadcast addresses for each IP address can also be specified; if they are not, then the cluster uses the netmask and broadcast addresses from the network interconnect in the subnet.
Device special file	Specify each shared disk partition used in a service.
File system and sharing options	If the service uses a file system, specify the type of file system, the mount point, and any mount options. You can specify any of the standard file system mount options as described in the mount(8) man page. It is not necessary to provide mount information for raw devices (if used in a service). ext2 and ext3 file systems are supported for a cluster. Specify whether to enable forced unmount for a file system. Forced unmount allows the cluster service management infrastructure to unmount a file system prior to relocation or failover, even if the file system is busy. This is accomplished by terminating any applications that are accessing the file system. You can also specify whether to export the file system via NFS set access permissions. Refer to Section 6.1 *Setting Up an NFS Service* for details. Specify whether or not to make the file system accessible to SMB clients via Samba by providing a Samba share name.

Table 4-1. Service Property and Resource Information

4.1.2. Creating Service Scripts

The cluster infrastructure starts and stops specified applications by running service-specific scripts. For both NFS and Samba services, the associated scripts are built into the cluster services infrastructure. Consequently, when running the **Cluster Configuration Tool** to configure NFS and Samba services, leave the **User Script** field blank. For other application types it is necessary to designate a service script. For example, when configuring a database application, specify the fully-qualified pathname of the corresponding database start script.

The format of the service scripts conforms to the conventions followed by the System V `init` scripts. This convention dictates that the scripts have a `start`, `stop`, and `status` clause. These scripts should return an exit status of `0` upon successful completion.

When a service fails to start on one cluster member, Red Hat Cluster Manager will attempt to start the service on other cluster members. If the other cluster members fail to start the service, Red Hat Cluster Manager attempts to stop the service on all members. If it fails to stop the service for any reason, the cluster infrastructure will place the service in the **Failed** state. Administrators must then start the **Cluster Status Tool**, highlight the failed service, and click **Disable** before they can enable the service.

In addition to performing the stop and start functions, the script is also used for monitoring the status of an application service. This is performed by calling the `status` clause of the service script. To enable service monitoring, specify a nonzero value in the **Check Interval** field when specifying service properties with the **Cluster Configuration Tool**. If a nonzero exit is returned by a status check request to the service script, then the cluster infrastructure first attempts to restart the application on the member it was previously running on. Status functions do not have to be fully implemented in service scripts. If no real monitoring is performed by the script, then a stub `status` clause should be present which returns success.

The operations performed within the status clause of an application can be tailored to best meet the application's needs as well as site-specific parameters. For example, a simple status check for a database would consist of verifying that the database process is still running. A more comprehensive check would consist of a database table query.

The `/usr/share/cluster/doc/services/examples/` directory contains a template that can be used to create service scripts, in addition to examples of scripts. Refer to Section 5.1 *Setting Up an Oracle Service*, Section 5.3 *Setting Up a MySQL Service*, Chapter 7 *Setting Up Apache HTTP Server*, for sample scripts.

4.1.3. Configuring Service Disk Storage

Prior to creating a service, set up the shared file systems and raw devices that the service is to use. Refer to Section 2.4.4 *Configuring Shared Disk Storage* for more information.

If employing raw devices in a cluster service, it is possible to use the `/etc/sysconfig/rawdevices` file to bind the devices at boot time. Edit the file and specify the raw character devices and block devices that are to be bound each time the member boots. Refer to Section 3.5 *Editing the `rawdevices` File* for more information.

Note that software RAID and host-based RAID cannot be used for shared disk storage. Only certified SCSI adapter-based RAID cards can be used for shared disk storage.

You should adhere to the following *service disk storage recommendations*:

- For optimal performance, use a 4 KB block size when creating file systems. Note that some of the `mkfs` file system build utilities default to a 1 KB block size, which can cause long `fsck` times.
- To facilitate quicker failover times, it is recommended that the ext3 file system be used. Refer to Section 2.4.4.6 *Creating File Systems* for more information.

- For large file systems, use the `nocheck` option to bypass code that checks all the block groups on the partition. Specifying the `nocheck` option can significantly decrease the time required to mount a large file system.

4.1.4. Verifying Application Software and Service Scripts

Prior to setting up a service, install any applications that are used in the service on each member in the cluster (or each member in the failover domain, if used). After installing the application on these members, verify that the application runs and can access shared disk storage. To prevent data corruption, do not run the application simultaneously on more than one member.

If using a script to start and stop the service application, install and test the script on all cluster members, and verify that it can be used to start and stop the application. Refer to Section 4.1.2 *Creating Service Scripts* for more information.

4.2. Displaying a Service Configuration

In the **Cluster Configuration Tool**, the following information for a given <**service**> element is viewable directly on the **Services** tab (when the service configuration is fully expanded):

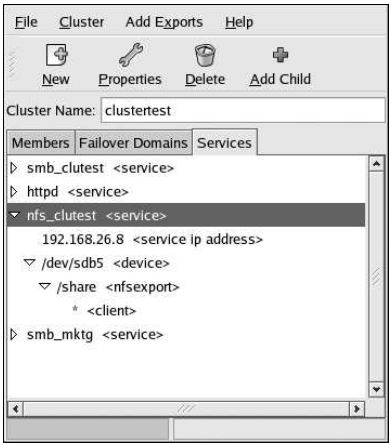


Figure 4-1. The Services Tab

- Service name
- Device special file name
- Service IP address
- NFS export and clients (if specified) for the device

You can display the failover domain, check interval, and user script name for a <**service**> by accessing its properties dialog box.

The Samba share name, mount point, file system type, and mount options for a service are displayed as the properties for a <**device**>.

You can view the permissions for NFS client access by displaying <**client**> properties.

To display the properties for an element in the cluster configuration, use any of the following methods:

- Double-click the element name.
- Select the element name and click **Properties**.
- Select the element name and choose **File => Properties**.

For instructions on configuring a service using the **Cluster Configuration Tool**, refer to Section 3.10 *Adding a Service to the Cluster*.

To display cluster service status, refer to Section 8.2 *Displaying Cluster and Service Status*.

4.3. Disabling a Service

A running service can be disabled to stop the service and make it unavailable. Once disabled, a service can then be re-enabled. Refer to Section 4.4 *Enabling a Service* for information.

There are several situations in which a running service may need to be disabled:

- To modify a service—A running service must be disabled before it can be modified. Refer to Section 4.5 *Modifying a Service* for more information.
- To temporarily stop a service—A running service can be disabled, making it unavailable to clients without having to completely delete the service.

To disable a running service, invoke the `redhat-config-cluster` command to display the **Cluster Status Tool**. In the **Services** tab, select the service you want to disable and click **Disable**.

4.4. Enabling a Service

When you enable a disabled service, that service is started and becomes available on the specified cluster member. (You can start the service on another member by dragging the service icon  into the **Members** tab of the **Cluster Status Tool** and dropping the icon on the member icon  .)

To enable a stopped service, invoke the `redhat-config-cluster` command to display the **Cluster Configuration Tool**. In the **Services** tab, select the service you want to enable and click **Enable**.

4.5. Modifying a Service

You can modify the properties for a service any time after creating the service. For example, you can change the IP address or location of the user script. You can also add more resources (for example, additional file systems or IP addresses) to an existing service. Refer to Section 4.1.1 *Gathering Service Information* for information.

Using the **Cluster Status Tool**, first disable, or stop, the service before modifying its properties. If an administrator attempts to modify a running service, a warning dialog displays prompting to stop the service. Refer to Section 4.3 *Disabling a Service* for instructions on disabling a service.

Because a service is unavailable while being modified, be sure to gather all the necessary service information before disabling it to minimize service down time. In addition, back up the cluster configuration file (`/etc/cluster.xml`) before modifying a service; refer to Section 8.5 *Backing Up and Restoring the Cluster Database*.

Use the **Cluster Configuration Tool** to modify the service properties, then save the configuration by choosing **File => Save**. After the modifications to the service have been saved, you can restart the service using the **Cluster Status Tool** to enable the modifications.

4.6. Relocating a Service

In addition to providing automatic service failover, a cluster enables you to cleanly stop a service on one member and restart it on another member. This service relocation functionality allows you to perform maintenance on a cluster member while maintaining application and data availability.

To relocate a service by using the **Cluster Status Tool**, drag the service icon  from the **Services** tab onto the member icon  in the **Members** tab. The Red Hat Cluster Manager stops the service on the member on which it was running and restarts it on the new member.

4.7. Deleting a Service

A cluster service can be deleted. Note that the cluster configuration file should be backed up before deleting a service. Refer to Section 8.5 *Backing Up and Restoring the Cluster Database* for information. To delete a service from the cluster, follow these steps:

1. If the service is running, disable the service using the **Cluster Status Tool** (refer to Section 4.3 *Disabling a Service*).
2. On the **Services** tab in the **Cluster Configuration Tool**, select the name of the service you want to remove and click **Delete**.
3. You are prompted to confirm the deletion. To remove the service, click **OK**.
4. Choose **File => Save** to save the change to the `/etc/cluster.xml` configuration file.

4.8. Handling Failed Services

The cluster puts a service into the **Failed** state if it is unable to successfully start the service across all members and then cannot cleanly stop the service. A **Failed** state can be caused by various problems, such as a misconfiguration as the service is running or a service hang or crash. The **Cluster Status Tool** displays the service as being **Failed**.

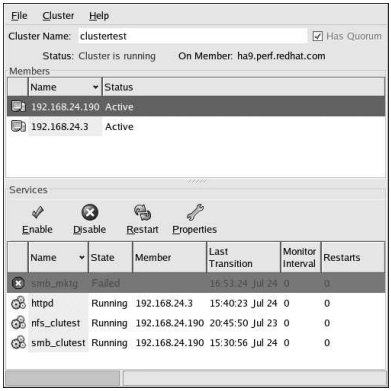


Figure 4-2. Service in Failed State



Note

You must disable a **Failed** service before you can modify or re-enable the service.

Be sure to carefully handle failed services. If service resources are still configured on the owner member, starting the service on another member may cause significant problems. For example, if a file system remains mounted on the owner member, and you start the service on another member, the file system is mounted on both members, which can cause data corruption. If the enable fails, the service remains in the **Disabled** state.

After highlighting the service and clicking **Disable**, you can attempt to correct the problem that caused the **Failed** state. After you modify the service, the cluster software enables the service on the owner member, if possible; otherwise, the service remains in the **Disabled** state. The following list details steps to follow in the event of service failure:

1. Modify cluster event logging to log debugging messages. Viewing the logs can help determine problem areas. Refer to Section 8.6 *Modifying Cluster Event Logging* for more information.
2. Use the **Cluster Status Tool** to attempt to enable or disable the service on one of the cluster or failover domain members. Refer to Section 4.3 *Disabling a Service* and Section 4.4 *Enabling a Service* for more information.
3. If the service does not start or stop on the member, examine the `/var/log/messages` and (if configured to log separately) `/var/log/cluster` log files, and diagnose and correct the problem. You may need to modify the service to fix incorrect information in the cluster configuration file (for example, an incorrect start script), or you may need to perform manual tasks on the owner member (for example, unmounting file systems).
4. Repeat the attempt to enable or disable the service on the member. If repeated attempts fail to correct the problem and enable or disable the service, reboot the member.
5. If still unable to successfully start the service, verify that the service can be manually restarted outside of the cluster framework. For example, this may include manually mounting the file systems and manually running the service start script.

Chapter 5.

Database Services

This chapter contains instructions for configuring Red Hat Enterprise Linux to make database services highly available.



Note

The following descriptions present example database configuration instructions. Be aware that differences may exist in newer versions of each database product. Consequently, this information may not be directly applicable.

5.1. Setting Up an Oracle Service

A database service can serve highly-available data to a database application. The application can then provide network access to database client systems, such as Web servers. If the service fails over, the application accesses the shared database data through the new cluster system. A network-accessible database service is usually assigned an IP address, which is failed over along with the service to maintain transparent access for clients.

This section provides an example of setting up a cluster service for an Oracle database. Although the variables used in the service scripts depend on the specific Oracle configuration, the example may aid in setting up a service for individual environments. Refer to Section 5.2 *Tuning Oracle Service* for information about improving service performance.

In the example that follows:

- The service includes one IP address for the Oracle clients to use.
- The service has two mounted file systems, one for the Oracle software (`/u01/`) and the other for the Oracle database (`/u02/`), which are set up before the service is added.
- An Oracle administration account with the name **oracle** is created on the cluster systems that run the service before the service are actually added.
- The administration directory is on a shared disk that is used in conjunction with the Oracle service (for example, `/u01/app/oracle/admin/db1`).

Create a consistent user/group configuration that can properly access Oracle service for each cluster system. For example:

```
mkdir /users
groupadd -g 900 dba
groupadd -g 901 oinstall
useradd -u 901 -g 901 -d /users/oracle -m oracle
usermod -G 900 oracle
```

The Oracle service example uses three scripts that must be placed in `/users/oracle` and owned by the Oracle administration account. The `oracle` script is used to start and stop the Oracle service. Specify this script when you add the service. This script calls the other Oracle example scripts. The `startdb` and `stopdb` scripts start and stop the database. Note that there are many ways for an application to interact with an Oracle database.

The following is an example of the `oracle` script, which is used to start, stop, and check the status of the Oracle service.

```
#!/bin/sh
#
# Cluster service script to start, stop, and check status of oracle
#

cd /users/oracle

case $1 in
start)
    su - oracle -c ./startdb
    ;;
stop)
    su - oracle -c ./stopdb
    ;;
status)
    status oracle
    ;;
esac
```

The following is an example of the `startdb` script, which is used to start the Oracle Database Server instance:

```
#!/bin/sh
#

#
# Script to start the Oracle Database Server instance.
#
######################################################################
#
# ORACLE_RELEASE
#
# Specifies the Oracle product release.
#
######################################################################

ORACLE_RELEASE=9.2.0

######################################################################
#
# ORACLE_SID
#
# Specifies the Oracle system identifier or "sid", which is the name of
# the Oracle Server instance.
#
######################################################################

export ORACLE_SID=TEST

######################################################################
#
# ORACLE_BASE
#
# Specifies the directory at the top of the Oracle software product and
# administrative file structure.
#
######################################################################

export ORACLE_BASE=/u01/app/oracle
```

```
#########################################################################
#
# ORACLE_HOME
#
# Specifies the directory containing the software for a given release.
# The Oracle recommended value is $ORACLE_BASE/product/<release>
#
#########################################################################

export ORACLE_HOME=/u01/app/oracle/product/${ORACLE_RELEASE}

#########################################################################
#
# LD_LIBRARY_PATH
#
# Required when using Oracle products that use shared libraries.
#
#########################################################################

export LD_LIBRARY_PATH=${ORACLE_HOME}/lib:$LD_LIBRARY_PATH

#########################################################################
#
# PATH
#
# Verify that the users search path includes $ORACLE_HOME/bin
#
#########################################################################

export PATH=$PATH:${ORACLE_HOME}/bin

#########################################################################
#
# This does the actual work.
#
# Start the Oracle Server instance based on the initSID.ora
# initialization parameters file specified.
#
#########################################################################

/u01/app/oracle/product/9.2.0/bin/sqlplus << EOF
sys as sysdba
spool /home/oracle/startdb.log
startup pfile = /u01/app/oracle/product/9.2.0/admin/test/scripts/init.ora open;
spool off
quit;
EOF

exit
```

The following is an example of the `stopdb` script, which is used to stop the Oracle Database Server instance:

```
#!/bin/sh
#
#
# Script to STOP the Oracle Database Server instance.
#
#########################################################################
#
# ORACLE_RELEASE
```

```
#
# Specifies the Oracle product release.
#
###########################################################################

ORACLE_RELEASE=9.2.0

###########################################################################
#
# ORACLE_SID
#
# Specifies the Oracle system identifier or "sid", which is the name
# of the Oracle Server instance.
#
###########################################################################

export ORACLE_SID=TEST

###########################################################################
#
# ORACLE_BASE
#
# Specifies the directory at the top of the Oracle software product
# and administrative file structure.
#
###########################################################################

export ORACLE_BASE=/u01/app/oracle

###########################################################################
#
# ORACLE_HOME
#
# Specifies the directory containing the software for a given release.
# The Oracle recommended value is $ORACLE_BASE/product/<release>
#
###########################################################################

export ORACLE_HOME=/u01/app/oracle/product/${ORACLE_RELEASE}

###########################################################################
#
# LD_LIBRARY_PATH
#
# Required when using Oracle products that use shared libraries.
#
###########################################################################

export LD_LIBRARY_PATH=${ORACLE_HOME}/lib:$LD_LIBRARY_PATH

###########################################################################
#
# PATH
#
# Verify that the users search path includes $ORACLE_HOME/bin
#
###########################################################################

export PATH=$PATH:${ORACLE_HOME}/bin

###########################################################################
#
```

```
# This does the actual work.
#
# STOP the Oracle Server instance in a tidy fashion.
#
######################################################################

/u01/app/oracle/product/9.2.0/bin/sqlplus << EOF
sys as sysdba
spool /home/oracle/stopdb.log
shutdown abort;
spool off
quit;
EOF

exit
```

5.1.1. Oracle and the Cluster Configuration Tool

To add an Oracle service using the **Cluster Configuration Tool**, perform the following:

1. Start the **Cluster Configuration Tool** by choosing **Main Menu => System Settings => Server Settings => Cluster** or by typing `redhat-config-cluster` at a shell prompt. The **Cluster Status Tool** appears by default.
2. Start the **Cluster Configuration Tool** by selecting **Cluster => Configure** from the **Cluster Status Tool** menus.
3. Click the **Services** tab.
4. Add the Oracle service.
   - Click **New**. The **Service** dialog appears.

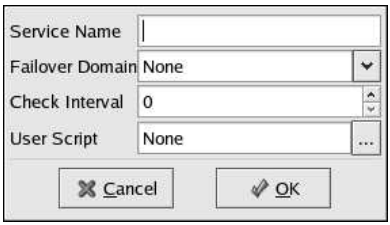


Figure 5-1. Adding an Oracle Service

   - Enter a **Service Name** for the Oracle service.
   - Select a **Failover Domain** or leave it as *None*.
   - Type a quantity (seconds) to check the health of the Oracle service through the `status` function of the init script.
   - Enter a **User Script**, such as `/home/oracle/oracle`.
   - Click **OK**

5. Add an IP address for the Oracle service.
   - Select the Oracle service and click **Add Child**.

- Select **Add Service IP Address** and click **OK**. The **Service IP Address** dialog appears.
- Enter an **IP Address**.
- Enter a **Netmask**, or leave it *None*.
- Enter a **Broadcast Address**, or leave it *None*.
- Click **OK**.

6. Add a device for the Oracle service and administrative files.
   - Select the Oracle service and click **Add Child**.
   - Select **Add Device** and click **OK**. The **Device** dialog appears.
   - Enter the **Device Special File** (for example, `/dev/sdb5`).
   - In the **Mount Point** field, enter **`/u01`**.
   - Select the file system type in **FS Type** or leave it blank.
   - Enter any mount point **Options**, including **`rw`** (read-write).
   - Check or uncheck **Force Unmount**.
   - Click **OK**.

7. Add a device for the Oracle database files.
   - Select the Oracle service and click **Add Child**.
   - Select **Add Device** and click **OK**. The **Device** dialog appears.
   - Enter the **Device Special File** (for example, `/dev/sdb6`).
   - In the **Mount Point** field, enter **`/u02`**.
   - Select the file system type in **FS Type** or leave it blank.
   - Enter any mount point **Options**, including **`rw`** (read-write).
   - Check or uncheck **Force Unmount**.
   - Click **OK**.

8. Choose **File => Save** to save the Oracle service.

5.2. Tuning Oracle Service

The Oracle database recovery time after a failover is directly proportional to the number of outstanding transactions and the size of the database. The following parameters control database recovery time:

- **LOG_CHECKPOINT_TIMEOUT**
- **LOG_CHECKPOINT_INTERVAL**
- **FAST_START_IO_TARGET**
- **REDO_LOG_FILE_SIZES**

To minimize recovery time, set the previous parameters to relatively low values. Note that excessively low values adversely impact performance. Try different values to find the optimal value.

Oracle provides additional tuning parameters that control the number of database transaction retries and the retry delay time. Be sure that these values are large enough to accommodate the failover time in the cluster environment. This ensures that failover is transparent to database client application programs and does not require programs to reconnect.

5.3. Setting Up a MySQL Service

A database service can serve highly-available data to a MySQL database application. The application can then provide network access to database client systems, such as Web servers. If the service fails over, the application accesses the shared database data through the new cluster system. A network-accessible database service is usually assigned one IP address, which is failed over along with the service to maintain transparent access for clients.

An example of a configuring a MySQL database service is as follows:

- The MySQL server packages are installed on each cluster system that will run the service. The MySQL database directory resides on a file system that is located on a disk partition on shared storage. This allows the database data to be accessed by all cluster members. In the example, the file system is mounted as `/var/lib/mysql`, using the shared disk partition `/dev/sda1`.
- An IP address is associated with the MySQL service to accommodate network access by clients of the database service. This IP address is automatically migrated among the cluster members as the service fails over. In the example below, the IP address is 10.1.16.12.
- The script that is used to start and stop the MySQL database is the standard `init` script `mysqld`. If general logging of connections and queries is needed, edit the `mysqld` script to add the option `--log=/var/log/mysqld.log` as the last option to the `safe_mysqld` command. The resulting line should appear similar to the following (Note: the forward slash (\) denotes the continuation of one line):

  ```
  /usr/bin/safe_mysqld  --defaults-file=/etc/my.cnf --log=/var/log/mysqld.log \
   >/dev/null 2>&1 &
  ```

  If the `--log` option is added to the `mysqld` script, the new `mysqld` script should be copied to the other cluster members that can run the MySQL service, so that they can log connections and queries if the MySQL service fails over to those members.
- by default, a client connection to a MySQL server times out after eight hours of inactivity. This connection limit can be modified by setting the `wait_timeout` variable in the `/etc/my.cnf` file. For example, to set timeouts to four hours, add the following line to the `[mysqld]` section of `/etc/my.cnf`:

  ```
  set-variable = wait_timeout=14400
  ```

  Restart the MySQL service. Note that after this change is made, the new `/etc/my.cnf` file should be copied to all the other cluster members that can run the MySQL service.

  To check if a MySQL server has timed out, invoke the `mysqladmin` command and examine the uptime. If it has timed out, invoke the query again to automatically reconnect to the server.

  Depending on the Linux distribution, one of the following messages may indicate a MySQL server timeout:

  ```
  CR_SERVER_GONE_ERROR
  CR_SERVER_LOST
  ```

5.3.1. MySQL and the Cluster Configuration Tool

To add a MySQL service using the **Cluster Configuration Tool**, perform the following:

1. Start the **Cluster Configuration Tool** by choosing **Main Menu => System Settings => Server Settings => Cluster** or by typing `redhat-config-cluster` at a shell prompt. The **Cluster Status Tool** appears by default.
2. Start the **Cluster Configuration Tool** by selecting **Cluster => Configure** from the **Cluster Status Tool** menus.
3. Click the **Services** tab.
4. Add the MySQL service.
   - Click **New**. The **Service** dialog appears.

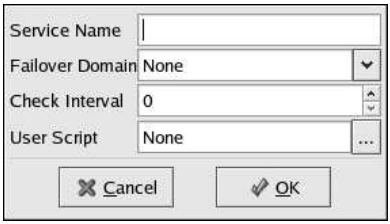


Figure 5-2. Adding a Service

   - Enter a **Service Name** for the MySQL service.
   - Select a **Failover Domain** or leave it as *None*.
   - Type a quantity (seconds) in the **Check Interval** box if you want to check the health of the MySQL service through the `status` directive of the mysqld init script.
   - Enter a **User Script**, such as `/etc/init.d/mysqld`.
   - Click **OK**.

5. Add an IP address for the MySQL service.
   - Select the MySQL service and click **Add Child**.
   - Select **Add Service IP Address** and click **OK**. The **Service IP Address** dialog appears.
   - Enter an **IP Address**.
   - Enter a **Netmask**, or leave it *None*.
   - Enter a **Broadcast Address**, or leave it *None*.
   - Click **OK**.

6. Add a device for the MySQL service.
   - Select the MySQL service and click **Add Child**.
   - Select **Add Device** and click **OK**. The **Device** dialog appears.
   - Enter the **Device Special File** (for example, `/dev/sdc3`).
   - In the **Mount Point** field, enter **`/var/lib/mysql`**.
   - Select the file system type in **FS Type** or leave it blank.
   - Enter any mount point **Options**, including **`rw`** (read-write).
   - Check or uncheck **Force Unmount**.

- Click **OK**.

7. Choose **File => Save** to save the MySQL service.

Chapter 6.

Network File Sharing Services

This chapter contains instructions for configuring Red Hat Enterprise Linux to make network file sharing services through NFS and Samba highly available.

6.1. Setting Up an NFS Service

A highly-available network file system (NFS) is one of the key strengths of the clustering infrastructure. Advantages of clustered NFS services include:

- Ensures that NFS clients maintain uninterrupted access to key data in the event of server failure.
- Facilitates planned maintenance by allowing transparent relocation of NFS services to one cluster member, allowing you to fix or upgrade the other cluster member.
- Allows setup of an active-active configuration to maximize equipment utilization. Refer to Section 6.5 *NFS Configuration: Active-Active Example* for more information.

6.1.1. NFS Server Requirements

To create highly available NFS services, there are a few requirements which must be met by each cluster member. (Note: these requirements do not pertain to NFS client systems.) These requirements are as follows:

- The NFS daemon must be running on all cluster servers. Check the status of the servers by running the following:
  ```
  /sbin/service nfs status
  ```
  NFS services will not start unless the following NFS daemons are running: `nfsd`, `rpc.mountd`, and `rpc.statd`. If the service is not running, start it with the following commands:
  ```
  /sbin/service portmap start
  /sbin/service nfs start
  ```
  To make NFS start upon reboot and when changing runlevels, run the following command:
  ```
  /sbin/chkconfig --level 345 nfs on
  ```
- The RPC `portmap` daemon must also be enabled with the following command:
  ```
  /sbin/chkconfig --level 345 portmap on
  ```
- File system mounts and their associated exports for clustered NFS services should *not* be included in `/etc/fstab` or `/etc/exports`. Rather, for clustered NFS services, the parameters describing mounts and exports are entered via the **Cluster Configuration Tool**. For your convenience, the tool provides a **Bulk Load NFS** feature to import entries from an existing file into the cluster configuration file.
- NFS *cannot* be configured to run over TCP with Red Hat Cluster Manager. For proper failover capabilities, NFS must run over the default UDP.

For detailed information about setting up an NFS server, refer to the *Red Hat Enterprise Linux System Administration Guide*.

6.2. Using the NFS Druid

This section describes how to use the **NFS Druid** to quickly configure an NFS share for client access.

1. Start the **Cluster Status Tool**. Verify that the cluster daemons are running; if not, choose **Cluster => Start Cluster Service** to start the cluster daemons.
2. In the **Cluster Status Tool**, choose **Cluster => Configure** to display the **Cluster Configuration Tool**.
3. Start the **NFS Druid** by choosing **Add Exports => NFS...** and click **Forward** to continue.

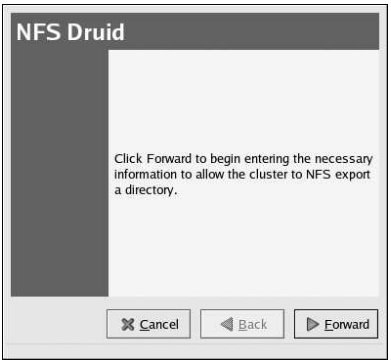


Figure 6-1. NFS Druid

4. Enter the **Export Directory** — Specified as a child of a device, the export directory can be the same as the mount point. In this case, the entire file system is accessible through NFS. Alternatively, you can specify a portion (subdirectory) of a mounted file system to be mounted (instead of the entire file system). By exporting subdirectories of a mountpoint, different access rights can be allocated to different sets of NFS clients.

   Enter the **Client Name** — Specified as a child of an export directory, the NFS client specification identifies which systems will be allowed to access the file system as NFS clients. You can specify individual systems (for example, **`fred`**) or groups of systems by using wildcards (for example, **`*.example.com`**). Entering an asterisk (**`*`**) in the **Client Name** field allows any client to mount the file system.

   Enter any **Client Options** in the provided fields — Specified as part of the **NFS Export Client** information, this field defines the access rights afforded to the corresponding client(s). Examples include **`ro`** (read only), and **`rw`** (read write). Unless explicitly specified otherwise, the default export options are **`ro`**,**`async`**,**`wdelay`**, **`root_squash`**. Refer to the `exports`(5) manpage for more options.



Figure 6-2. Export and Client Options

5. If an existing service contains the device and mountpoint configuration for the directory you want to NFS export, then select that existing service. Otherwise, enter a new **Service Name** and **Service IP Address** for the NFS export directory.

   Service Name — A name used to uniquely identify this service within the cluster (such as **`nfs_cluster`** or **`marketing`**.)

   Service IP Address — NFS clients access file systems from an NFS server which is designated by its IP address (or associated hostname). To keep NFS clients from knowing which specific cluster member is the acting NFS server, the client systems should not use the cluster member's hostname as the IP address from which a service is started. Rather, clustered NFS services are assigned floating IP addresses which are distinct from the cluster server's IP addresses. This floating IP address is then configured on whichever cluster member is actively serving the NFS export. Following this approach, the NFS clients are only aware of the floating IP address and are unaware of the fact that clustered NFS server has been deployed.



Figure 6-3. Select Service for Export

6. For non-clustered file systems, the mount information is typically placed in `/etc/fstab`. However, clustered file systems must not be placed in `/etc/fstab`. This is necessary to ensure that

only one cluster member at a time has the file system mounted. Failure to do so will likely result in file system corruption and system crashes.

If you selected an existing service, then devices for that service will be listed under **Existing Device and Mountpoint**. If the device and mount point for your NFS export is listed, then select it.

Otherwise, select **New Device** and use the fields to edit the following settings.

Device Special File — Designates the disk or partition on shared storage.

Device Mountpoint — Specifies the directory on which the file system will be mounted. An NFS service can include more than one file system mount. In this manner, the file systems will be grouped together as a single failover unit.



Figure 6-4. Select Device for Export

7. At the end of the **NFS Druid**, click **Apply** to create the service. Save the configuration by choosing **File => Save** from the **Cluster Configuration Tool**.

To modify your NFS service configuration, click the **Services** tab in the **Cluster Configuration Tool** and click the triangular icon ▷ next to the NFS service to display the full child tree for the service. Double-click each child to modify options.

1. Highlight the <**service**> and click **Properties** to configure the the following options:

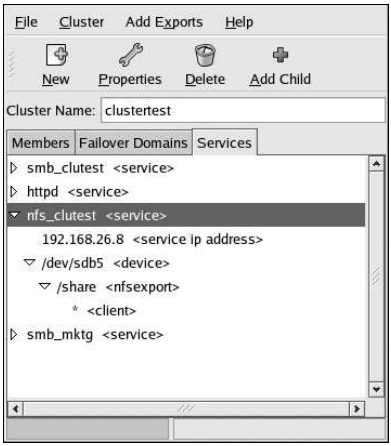


Figure 6-5. Services in the Cluster Configuration Tool

- **Service Name** — A name used to uniquely identify this service within the cluster (such as **`nfs_cluster`** or **`marketing`**.
- **Failover Domain** — An optional property that specifies a subset (or ordered subset) of cluster members which are eligible to run the service in the event of a failover. You must create the failover domain before you can reference it in an NFS service configuration; see Section 3.9 *Configuring a Failover Domain* for more information.
- **Check Interval** — An optional property that specifies whether or not to check the status of the NFS daemons at a regular interval (in seconds). The default value is 0 seconds, meaning the daemon status is not checked.

  If the service returns an error or does not respond to the status check, the cluster attempts to cleanly shut down the service and start it on another member. If at any point it fails to cleanly shut down the NFS service, the cluster will place the service in a **Failed** state, requiring the administrator to disable the service first before attempting to restart it.
- For the **User Script**, leave the field as **`None`**, as the cluster infrastructure handles NFS service control and status checking.

2. Choose the <**service ip address**> child to change the **Service IP Address** and to enter a **Netmask** and **Broadcast** address, which are both set as **`None`** by default. If these fields are left as **`None`**, then the cluster infrastructure will use the netmask and broadcast IP address configured on the network device of the member running the service.
3. Choose the <**device**> child to modify the **Device Special File**, **Device Mount Point**, **FS Type**, and **Mount Options**. You can also check or uncheck the **Force Unmount**. When **Forced Unmount** is enabled, any applications that have the specified file system mounted will be killed prior to disabling or relocating the NFS service (assuming the application is running on the same member that is running the NFS service)
4. Choose the <**nfsexport**> child to specify a directory name for clients to mount the exported share.

5. Choose the <**client**> child to enter **Client Name**, any hosts, groups, and domains that are allowed to mount the exported shares (default is ***** which allows any client to mount the share) and **Options** for allowed client mount options (such as **rw** for read-write or **ro** for read-only).

6.2.1. NFS Client Access

The NFS usage model for clients is completely unchanged from its normal approach. For example, to mount the NFS share from `clu1.example.com` to the client's `/mnt/users/` directory, run the following command:

```
/bin/mount -t nfs clu1.example.com:/share /mnt/users
```

To simplify the mounting of the NFS share for clients, place the following in the client's `/etc/fstab` file:

```
clu1.example.com:/share /mnt/users nfs rw,rsize=8192,wsize=8192 0 0
```

For additional NFS mount options, refer to the *Red Hat Enterprise Linux System Administration Guide*.

6.3. NFS Caveats

The following points should be taken into consideration when clustered NFS services are configured.

Avoid using `exportfs -r`

File systems being NFS exported by cluster members do not get specified in the conventional `/etc/exports` file. Rather, the NFS exports associated with cluster services are specified in the cluster configuration file (as established by the **Cluster Configuration Tool**).

The command `exportfs -r` removes any exports which are not explicitly specified in the `/etc/exports` file. Running this command causes the clustered NFS services to become unavailable until the service is restarted. For this reason, it is recommended to avoid using the `exportfs -r` command on a cluster on which highly available NFS services are configured. To recover from unintended usage of `exportfs -r`, the NFS cluster service must be stopped and then restarted.

NFS File Locking

NFS file locks are *not* preserved across a failover or service relocation. This is due to the fact that the Linux NFS implementation stores file locking information in system files. These system files representing NFS locking state are not replicated across the cluster. The implication is that locks may be regranted subsequent to the failover operation.

6.4. Importing the Contents of an NFS Exports File

The **Bulk Load NFS** feature allows you to import all the entries from an `/etc/exports` file without having to create the entries individually as children of a device. This can be convenient for administrators who are transitioning from traditional NFS server systems to a high-availability cluster. To use the **Bulk Load NFS** feature, follow these steps:

1. Stop all non-clustered NFS exports (for example, `/sbin/service nfs stop`).

2. Unmount the file systems on which the exports reside (for example, `/bin/umount /mnt/nfs/`, where */mnt/nfs* is the directory on which the NFS exported partitions are mounted).
3. Start the **Cluster Configuration Tool**.
4. On the **Services** tab, select the device(s) to be loaded by the `/etc/exports` file.
5. Choose **File => Bulk Load NFS**.
6. Select the exports file that you want to load into the selected device (by default, the **Bulk Load NFS** dialog attempts to load `/etc/exports`), and click **OK**.

   All entries in the file specified in the **Bulk Load NFS** dialog box are subjected to the same validation as when an NFS export directory is manually specified. Any entries in the imported file that fail validation are displayed in a message box. Only entries that pass validation are loaded into the selected device.
7. If the bulk load successfully completed, you must then remove all exports from the `/etc/exports` file so that it does not affect the cluster NFS services.
8. Choose **File => Save** to save the change to the `/etc/cluster.xml` configuration file.

6.5. NFS Configuration: Active-Active Example

Section 6.2 *Using the **NFS Druid*** described how to configure a simple NFS service using the **NFS Druid**. This section shows how to configure a second NFS service on another running cluster member. The second service has its own separate IP address and failover domain. This cluster configuration, called an *active-active configuration*, allows multiple cluster members to simultaneously export file systems. This most effectively utilizes the capacity of cluster. In the event of a failure (or planned maintenance) on any cluster member running NFS services, those services will failover to the active cluster member.

For this example, individual subdirectories of the mounted file system will be made accessible on a read-write (**`rw`**) basis by three members of a department. The names of the systems used by these three team members are `ferris`, `denham`, and `brown`. To make this example more illustrative, notice that each team member will only be able to NFS mount their specific subdirectory, which has already been created for them and over which they have user and group permissions.

Use the **Cluster Configuration Tool** as follows to configure this example NFS service:

1. Verify that the cluster daemons are running in the **Cluster Status Tool**; if not, choose **Cluster => Start Local Cluster Daemons**.
2. Choose **Cluster => Configure** to display the **Cluster Configuration Tool**.
3. Choose the **Services** tab and, if services have already been defined, select one and click **New**. (If no services are defined, just click **New**.)
   a. Specify **`nfs_engineering`** in the **Service Name** field. This name was chosen as a reminder of the service's intended function to provide exports to the members of the engineering department.
   b. In this example, assume a failover domain named `clu3_domain` was previously created using the **Cluster Configuration Tool**, consisting only of member `clu3` with both **Restricted failover to only these members** and **Ordered Failover** unchecked. In this way, `clu3` is designated as the preferred member for this service. (Note that the `nfs_accounting` service is assigned to failover domain `clu4_domain`.) Choose **`clu3_domain`** from the **Failover Domain** list. (For more information on failover domains, refer to Section 3.9 *Configuring a Failover Domain*.)

c. Specify a value of **30** in the **Check Interval** field to specify that the status of the NFS daemons should be checked every 30 seconds.

d. The cluster infrastructure includes support for NFS services. Consequently, there is no need to create or specify a value for **User Script** when configuring an NFS service. Accept the default value of **None**.

e. Click **OK** to complete this portion of the service configuration.

4. In the **Cluster Configuration Tool**, select the service you just created, and click **Add Child**. On the **Add Device or IP Address** dialog box, choose **Add Service IP Address** and click **OK**.

   a. In the **Service IP Address** field, enter **10.0.0.11**. This example assumes a hostname of clunfseng is associated with this IP address, by which NFS clients mount the file system. Note that this IP address must be distinct from that of any cluster member.

   b. The default netmask address will be used, so accept the default of **None**.

   c. The default broadcast address will be used, so accept the default of **None**.

   d. Click **OK** to complete the service IP address configuration.

5. In the **Cluster Configuration Tool**, select the nfs_engineering service and click **Add Child**. On the **Add Device or IP Address** dialog box, choose **Add Device** and click **OK**.

   a. In the **Device Special File** field, enter **/dev/sdb11** which refers to the partition on the shared storage RAID box on which the file system will be physically stored.

   Leave the **Samba Share Name** field blank.

   b. In the **Mount Point** field, enter **/mnt/users/engineering**.

   c. From the **FS Type** menu, choose **ext3**.

   d. In the **Options** field, enter **rw,nosuid,sync**.

   e. Leave the **Force Unmount** checkbox checked.

   f. Click **OK** to complete this portion of the device configuration.

6. In the **Cluster Configuration Tool**, select the device you just created, and click **Add Child**.

   Enter **/mnt/users/engineering/ferris** in the **NFS Export Directory Name** field and click **OK**.

   Repeat this step twice, adding NFS export directories named **/mnt/users/engineering/denham** and **/mnt/users/engineering/brown**.

7. In the **Cluster Configuration Tool**, select the NFS export for ferris and click **Add Child**. The **NFS Export Client** dialog box is displayed.

   In the **Client Name** field, type **ferris**.

   In the **Options** field, type **rw**.

   Click **OK**.

8. Repeat step 7 twice, specifying clients named **denham** and **brown**, respectively, each with the same permissions options (**rw**).

9. Save the service by selecting **File => Save** in the **Cluster Configuration Tool**.

10. Start the service from the **Cluster Status Tool** by highlighting the service and clicking **Start**.

6.6. Setting Up a Samba Service

Highly available network file services are one of the key strengths of the clustering infrastructure. Advantages of high availibility Samba services include:

- Heterogeneous file serving capabilities to Microsoft® Windows™ clients using the CIFS/SMB protocol.
- Allows the same set of file systems to be simultaneously network served to both NFS and Windows based clients.
- Ensures that Windows-based clients maintain access to key data, and can quickly reestablish connection in the event of server failure.
- Facilitates planned maintenance by allowing the transparent relocation of Samba services to one cluster member, enabling you to fix or upgrade the other cluster member.
- Allows the setup of an active-active configuration to maximize equipment utilization.



Note

A complete explanation of Samba configuration is beyond the scope of this document. Rather, this documentation highlights aspects which are crucial for clustered operation. Refer to *Red Hat Enterprise Linux System Administration Guide* for more details on Samba configuration.

6.6.1. Samba Server Requirements

If you intend to create highly available Samba services, each cluster member on which the services will run must meet the following requirements:

- The Samba RPM packages must be installed. Note that there have been no modifications to the Samba RPMs to support high availability.
- The Samba daemons will be started and stopped by the cluster infrastructure on a per-service basis. Consequently, the Samba configuration information should *not* be specified in the conventional `/etc/samba/smb.conf` file. The **Cluster Configuration Tool** writes a `smb.conf.`*`sharename`* file to the `/etc/samba/` directory for each Samba share (where *`sharename`* is the name you specified for the Samba share).
- The automated system startup of the Samba daemons `smbd` and `nmbd` should be disabled in `init.d` runlevels. For example: `chkconfig --del smb`.
- Since the cluster infrastructure stops the cluster-related Samba daemons appropriately, do not manually run the conventional Samba stop script (`service smb stop`) as this will terminate all cluster-related samba daemons.
- File system mounts for clustered Samba services should not be included in `/etc/fstab`. Rather, for clustered services, the parameters describing mounts are entered via the **Cluster Configuration Tool**.
- Failover of Samba printer shares is not currently supported.

6.6.2. Samba Operating Model

This section provides background information describing the implementation model in support of Samba high availability services. Knowledge of this information will provide the context for understanding the configuration requirements of clustered Samba services.

The conventional, non-clustered Samba configuration model consists of editing the `/etc/samba/smb.conf` file to designate which file systems are to be made network accessible to the specified clients. It also designates access permissions and other mapping capabilities. In the single system model, a single instance of each of the `smbd` and `nmbd` daemons are automatically started up by the `/etc/rc.d/init.d/smb` runlevel script.

To implement high availibility Samba services, rather than having a single `/etc/samba/smb.conf` file, each service has its own Samba configuration file. These files are named `/etc/samba/smb.conf.`*`sharename`*, where *`sharename`* is the specific name of the individual configuration file associated with a Samba service. For example, if you created a share called *`mktg`*, the corresponding Samba configuration file would be `/etc/samba/smb.conf.`*`mktg`*.



Note

A Samba share must be in a service with *at least one* IP address.

The format of the `smb.conf.`*`sharename`* file is identical to the conventional `smb.conf` format. No additional fields have been created for clustered operation. There are several fields within the `smb.conf.`*`sharename`* file which are required for correct cluster operation; these fields will be described in Section 6.8 *Fields in the `smb.conf.`sharename File*. When a new Samba service is created using the **Cluster Configuration Tool**, the corresponding `smb.conf.`*`sharename`* file is created based on the service-specific parameters, including appropriate client systems, specific directories to share, and read-write permissions.

Copy the `/etc/samba/smb.conf.`*`sharename`* files onto all members in the cluster (or all members in an unrestricted failover domain, if used); refer to Section 3.9 *Configuring a Failover Domain* for more information. After the initial configuration, should any changes be made to any `smb.conf.`*`sharename`* file, you must also copy the updated version to the other members.

To facilitate high-availability Samba functionality, each individual Samba service configured within the cluster (through the **Cluster Configuration Tool**) will have its own individual pair of `smbd` and `nmbd` daemons. Consequently, if there are more than one Samba services configured with the cluster, you may see multiple instances of these daemon pairs running on an individual cluster server. These Samba daemons `smbd` and `nmbd` are not initiated via the conventional `init.d` run level scripts; rather they are initiated by the cluster infrastructure based on whichever member is the active service provider.

To allow a single system to run multiple instances of the Samba daemons, every pair of daemons is required to have both its own locking directory and its own process ID (pid) directory. Consequently, there will be a separate per-service Samba daemon locking and running process directory. These directories are given the name `/var/cache/samba/`*`sharename`*`/` for lock files and `/var/run/samba/`*`sharename`*`/` for pid files (where *`sharename`* is replaced by the Samba share name specified within the service configuration information set using the **Cluster Configuration Tool**). Continuing the prior example, the corresponding directories for our *`mktg`* share would be `/var/cache/samba/`*`mktg`*`/` and `/var/run/samba/`*`mktg`*`/`.

6.7. Using the Samba Druid

This section describes how to use the **Samba Druid** to quickly configure an Samba share for client access.

1. Start the **Cluster Status Tool**. Verify that the cluster daemons are running; if not, choose **Cluster => Start Cluster Service** to start the cluster daemons.
2. In the **Cluster Status Tool**, choose **Cluster => Configure** to display the **Cluster Configuration Tool**.
3. Start the **Samba Druid** by choosing **Add Exports => Samba...** and click **Forward** to continue.

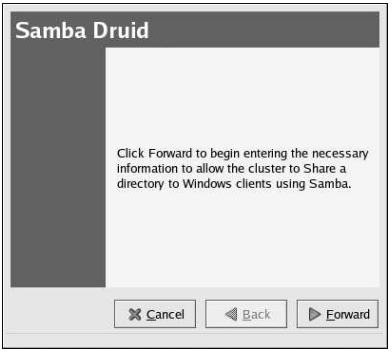


Figure 6-6. Samba Druid

4. Choose to create a new service with a new floating IP address.

   Enter a **Service Name** — A name used to uniquely identify this service within the cluster.

   Enter a **Service IP Address** — Clients access file shares from a server as designated by its IP address (or associated hostname). To abstract clients from knowing which specific cluster member is the acting Samba server, the client systems should not use the cluster member's hostname as the IP address by which a service is accessed. Rather, clustered Samba services are assigned floating IP addresses which are distinct from the cluster server's IP addresses. This floating IP address is then configured on which ever cluster member is actively serving the share. Following this approach, the clients are only aware of the floating IP address and are unaware of the fact that clustered Samba services have been deployed.



Figure 6-7. Select Service for Export

5. Enter the device special filename and mount point for the service.

   Mount information — For non-clustered file systems, the mount information is typically placed in `/etc/fstab`. In contrast, clustered file systems must not be placed in `/etc/fstab`. This is necessary to ensure that only one cluster member at a time has the file system mounted. Failure to do so will result in file system corruption and potential system crashes.

   - **Device Special File** — The mount information designates the disk's device special file and the directory on which the file system will be mounted.
   - **Device Mount point** — A Samba service can include more than one file system mount. In this manner, the file systems will be grouped together as a single failover unit.



Figure 6-8. Select Device for Export

6. Enter a **Share Name** — Specifies the name by which clients refer to the mount point. Based on the name you specify, a corresponding `/etc/samba/smb.conf.`*`sharename`* file and lock directory `/var/cache/samba/`*`sharename`* will be created. By convention the actual Windows share name specified within the `smb.conf.`*`sharename`* will be set in accordance with this parameter. In practice, you can designate more than one Samba share within an individual `smb.conf.`*`sharename`* file. There can be at most one samba configuration specified per

service, which must be specified with the first device. For example, if you have multiple disk devices (and corresponding file system mounts) within a single service, then specify a single *sharename* for the service. Then within the `/etc/samba/smb.conf.`*sharename* file, designate multiple individual Samba shares to share directories from the multiple devices. To disable Samba sharing of a service, the share name should be set to **None**.

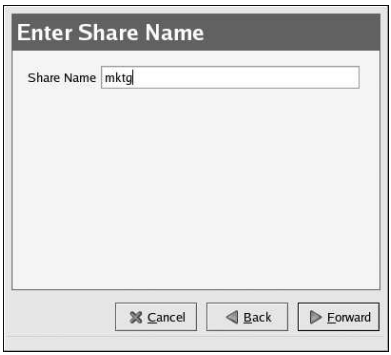


Figure 6-9. Samba Share Name

7. Click **Apply** to save the configurationg file (`/etc/samba/smb.conf.`*sharename*) to the cluster member.

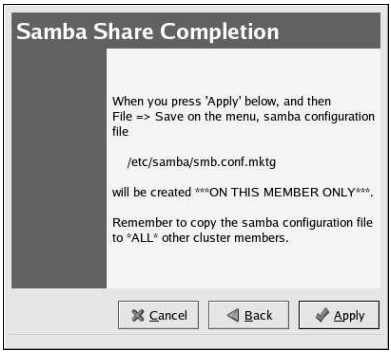


Figure 6-10. Samba Share Completion

8. Save the configuration for the Samba service by choosing **File => Save** from the **Cluster Configuration Tool**.
9. Copy `/etc/samba/smb.conf.`*sharename* over to the other cluster members.

To modify your Samba service configuration, click the **Services** tab in the **Cluster Configuration Tool** and click the triangular icon ▷ next to the Samba service to display the full child tree for the service. Double-click each child to modify options.

- Highlight the <**service**> and click **Properties** to configure the following options:

Service Name — A name used to uniquely identify this service within the cluster.

Failover Domain — Defines which systems are eligible to be the Samba server for this service when more than one cluster member is operational.

Check Interval — Specifies how often (in seconds) the cluster subsystem should verify that the Samba daemons (`smbd` and `nmbd`) associated with this service are running. In the event that either of these daemons have unexpectedly exited, they will be automatically restarted to resume services. If a value of 0 is specified, then no monitoring will be performed. For example, designating an interval of 90 seconds will result in monitoring at that interval.

For the **User Script**, leave the field as **None**, as the cluster infrastructure handles NFS service control and status checking.

- Choose the <**service ip address**> child to change the **Service IP Address** and to enter a **Netmask** and **Broadcast** address, which are both set as **None** by default. If these fields are left as **None**, then the cluster infrastructure will use the netmask and broadcast IP address configured on the network device of the member running the service.
- Choose the <**device**> child to modify the **Device Special File** and **Samba Share Name**, **Mount Point**, **FS Type**, and **Mount Options**.

  You can also check or uncheck the **Force Unmount** button. As part of the mount information, you can specify whether forced unmount should be enabled or not. When forced unmount is enabled, if any applications running on the cluster server have the designated file system mounted when the service is being disabled or relocated, then that application will be killed to allow the unmount to proceed.

6.7.1. Samba Considerations

When running the **Cluster Configuration Tool** to configure Samba services:

- Correctly enter the service parameters, as the validation logic associated with Samba parameters is not robust.
- After configuring a Samba service via the **Cluster Configuration Tool**, remember to tune the `/etc/samba/smb.conf.`*`sharename`* file for each service in accordance with the clients and authorization scheme you desire.
- Remember to copy the `smb.conf.`*`sharename`* file over to other cluster members.
- Note that the cluster infrastructure creates Samba lock directories when it starts the service.
- If you delete a Samba service, the **Cluster Configuration Tool** automatically deletes this file to preserve your site-specific configuration parameters for possible later use.

6.8. Fields in the `smb.conf.`*`sharename`* File

This section describes the fields within the `smb.conf.`*`sharename`* file which are most relevant to the correct operation of highly available Samba services. It is beyond the scope of this document to completely describe all of the fields within a Samba configuration file. There have been no additional field names added in support of clustering, and the file format follows the normal Samba conventions.

The following is an example `smb.conf.`*`sharename`* file which is automatically generated by the **Cluster Configuration Tool**. The example shows a share named `mktg`. The name of the file is `/etc/samba/smb.conf.mktg`.

```
# Template samba service configuration file - please modify
# to specify subdirectories and client access permissions.
```

```
# Remember to copy this file over to *ALL* other cluster
# members.
#
# From a cluster perspective, the key fields are:
# lock directory - must be unique per samba service.
# bind interfaces only - must be present set to yes.
# interfaces - must be set to service floating IP address.
# path - must be the service mountpoint or subdirectory thereof.
# Refer to the cluster documentation for details.

[global]
 workgroup = RHCLUSTER
 pid directory = /var/run/samba/mktg
 lock directory = /var/cache/samba/mktg
 log file = /var/log/samba/%m.log
 encrypt passwords = yes
 bind interfaces only = yes
 interfaces = 192.168.26.11

[mktg]
 comment = High Availability Samba Service
 browsable = yes
 writable = no
 public = yes
 path = /share
```

The following are descriptions of the most relevant fields, from a clustering perspective, in the `/etc/samba/smb.conf.`*`sharename`* file. In this example, the file is named `/etc/samba/smb.conf.mktg` in accordance with the share name that was specified (**`mktg`**) while running the **Cluster Configuration Tool**. Only the cluster-specific fields are described below. The remaining fields follow standard Samba syntax.

Global Parameters

These parameters pertain to all shares which are specified in the `smb.conf.`*`sharename`* file. It is possible to designate more than one share within this file, provided that the directories described within it are within the service's file system mounts.

`lock directory`

Dictates the name of the directory in which the Samba daemons (`smbd` and `nmbd`) will place their locking files. This must be set to `/var/cache/samba/`*`sharename`*, where *`sharename`* varies based on the parameter specified in the **Cluster Configuration Tool**. Specification of a lock directory is required to allow a separate per-service instance of `smbd` and `nmbd`.

`pid directory`

Dictates the name of the directory in which the Samba daemons (`smbd` and `nmbd`) will place their processor ID (pid) files. This must be set to `/var/run/samba/`*`sharename`*`/`, where *`sharename`* varies based on the parameter specified in the **Cluster Configuration Tool** tool. Specification of a pid directory is required to allow a separate per-service instance of `smbd` and `nmbd`.

`bind interfaces only`

This parameter must be set to *`yes`* to allow each `smbd` and `nmbd` pair to bind to the floating IP address associated with this clustered Samba service.

interfaces

Specifies the IP address associated with the Samba service. If a netmask is specified within the service, this field appears like the following example: *interfaces = 10.0.0.10/255.255.254.0*

Share-specific parameters

These parameters pertain to a specific Samba share.

writable

By default, the share access permissions are conservatively set as non-writable. Tune this parameter according to your site-specific preferences.

path

Defaults to the first file system mount point specified within the service configuration. This should be adjusted to match the specific directory or subdirectory intended to be available as a share to clients.

Chapter 7.

Setting Up Apache HTTP Server

This chapter contains instructions for configuring Red Hat Enterprise Linux to make the Apache HTTP Server highly available.

It provides an example of setting up a cluster service that fails over an Apache HTTP Server. Although the actual variables used in the service depend on the specific configuration, the example may assist in setting up a service for a particular environment.

7.1. Apache HTTP Server Setup Overview

First, configure Apache HTTP Server on all members in the cluster. Consider assigning the service to a failover domain to reduce the number of systems that must be configured to run this service. Refer to Section 3.9 *Configuring a Failover Domain* for instructions. The cluster software ensures that only one cluster system runs the Apache HTTP Server at one time. The configuration consists of installing the `httpd` RPM package on all cluster members (or on members in the failover domain, if used) and configuring a shared file system to house the website's content.

When installing the Apache HTTP Server on the cluster systems, do not configure the cluster systems so that the service automatically starts when the system boots by performing the following command:

```
chkconfig --del httpd
```

Rather than having the system startup scripts spawn `httpd`, the cluster infrastructure does that on the active cluster server. This ensures that the corresponding IP address and file system mounts are active on only one cluster member at a time.

When adding an `httpd` service, a *floating* IP address must be assigned to the service such that the IP address will transfer from one cluster member to another in the event of failover or service relocation. The cluster infrastructure binds this IP address to the network interface on the cluster system that is currently running the Apache HTTP Server. This IP address ensures that the cluster system running the `httpd` service is transparent to the HTTP clients accessing the Apache HTTP Server.

The file systems that contain the Web content must not be automatically mounted on shared disk storage when the cluster systems boot. Instead, the cluster software must mount and unmount the file systems as the `httpd` service is started and stopped on the cluster systems. This prevents the cluster systems from accessing the same data simultaneously, which may result in data corruption. Therefore, do not include the file systems in the `/etc/fstab` file.

7.2. Configuring Shared Storage

To set up the shared file system, perform the following tasks as root on one cluster system:

1. On a shared disk, use the interactive `parted` utility to create a partition to use for the document root directory. Note that it is possible to create multiple document root directories on different disk partitions. Refer to Section 2.4.4.4 *Partitioning Disks* for more information.
2. Use the `mkfs` command to create an ext3 file system on the partition you created in the previous step. Specify the drive letter and the partition number. For example:
   `mkfs -t ext3 /dev/sde3`
3. Mount the file system that contains the document root directory. For example:
   `mount /dev/sde3 /var/www/html`

Do not add this mount information to the `/etc/fstab` file because only the cluster software can mount and unmount file systems used in a service.

4. Copy all the required files to the document root directory.
5. If you have CGI files or other files that must be in different directories or in separate partitions, repeat these steps, as needed.

7.3. Installing and Configuring the Apache HTTP Server

The Apache HTTP Server must be installed and configured on all members in the assigned failover domain, if used, or in the cluster. The basic server configuration must be the same on all members on which it runs for the service to fail over correctly. The following example shows a basic Apache HTTP Server installation that includes no third-party modules or performance tuning.

On all member systems in the cluster (or members in the failover domain, if used), install the `httpd` RPM package. For example:

```
rpm -Uvh httpd-<version>.<arch>.rpm
```

On one system, perform the following tasks:

1. Edit the `/etc/httpd/conf/httpd.conf` configuration file and customize the file according to your configuration. For example:
   - Specify the directory that contains the HTML files. Also specify this mount point when adding the service to the cluster configuration. It is only required to change this field if the mountpoint for the website's content differs from the default setting of `/var/www/html/`. For example:
     ```
     DocumentRoot "/mnt/httpdservice/html"
     ```
   - If the script directory resides in a non-standard location, specify the directory that contains the CGI programs. For example:
     ```
     ScriptAlias /cgi-bin/ "/mnt/httpdservice/cgi-bin/"
     ```
   - Specify the path that was used in the previous step, and set the access permissions to default to that directory. For example:
     ```
     <Directory /mnt/httpdservice/cgi-bin">
     AllowOverride None
     Options None
     Order allow,deny
     Allow from all
     </Directory>
     ```

     Additional changes may need to be made to tune the Apache HTTP Server or add module functionality. For information on setting up other options, refer to the *Red Hat Enterprise Linux System Administration Guide* and the *Red Hat Enterprise Linux Reference Guide*.

2. The standard Apache HTTP Server start script, `/etc/rc.d/init.d/httpd` is also used within the cluster framework to start and stop the Apache HTTP Server on the active cluster member. Accordingly, when configuring the service, specify this script in the **User Script** field of the **Service** dialog box.
3. Copy the configuration file over to the other members of the cluster (or members of the failover domain, if configured).

Before the service is added to the cluster configuration, ensure that the Apache HTTP Server directories are not mounted. Then, on one member, invoke the **Cluster Configuration Tool** to add the service, as follows. This example assumes a failover domain named `httpd-domain` was created for

this service. Figure 7-1 shows the <**device**> settings for configuring the shared storage and mount point where HTML files and CGI scripts will be stored.

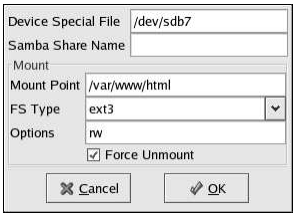


Figure 7-1. Configuring Apache HTTP Server

1. Select the **Services** tab and click **New**. The **Service** properties dialog box is displayed.
   a. Give the service a name (for example, **`httpd`**).
   b. Choose **`httpd-domain`** from the **Failover Domain** list.
   c. Specify a value in the **Check Interval** field.
   d. Specify **`/etc/rc.d/init.d/httpd`** in the **User Script** field.
   e. Click **OK**.

2. Select the `httpd` service on the **Services** tab and click **Add Child**. The **Add Device or Service IP Address** dialog box is displayed.
   a. Choose **Add Device** and click **OK**. The **Device** properties dialog box is displayed.
   b. Enter the device special file name in the **Device Special File** field (for example, **`/dev/hda7`**).
   c. Enter the mount point in the **Mount Point** field (for example, **`/var/www/html/`**).
   d. Choose **`ext3`** from the **FS Type** list.
   e. Enter **`rw`** in the **Options** field.
   f. Ensure that **Force Unmount** is checked, and click **OK**.

3. Ensure that the `httpd` service is still selected in the **Services** tab and click **Add Child**. The **Add Device or Service IP Address** dialog box is displayed.
   a. Choose **Add Service IP Address** and click **OK**. The **Service IP Address** properties dialog box is displayed.
   b. In the **IP Address** field, specify an IP address, which the cluster infrastructure binds to the network interface on the cluster system that runs the `httpd` service (for example, **`192.168.26.10`**).
   c. Specify a netmask of **`None`** in the **Netmask** field.
   d. In the **Broadcast** field, specify an IP address of **None** for broadcasting on the cluster subnet.
   e. Click **OK**.

4. Choose **File => Save** to save your changes.
5. To start the Apache HTTP Server within the **Cluster Status Tool**, highlight the service and click **Enable**.

Chapter 8.

Cluster Administration

This chapter describes the various administrative tasks involved in maintaining a cluster after it has been installed and configured.

8.1. Overview of the Cluster Status Tool

The **Cluster Status Tool** displays the status of the cluster service, cluster members, and application services, and shows statistics concerning service operation.

The cluster configuration file (maintained by the **Cluster Configuration Tool**) is used to determine how to manage the members, services, and cluster daemons.



Important

Do not manually edit the cluster configuration file.

Use the **Cluster Status Tool** to start and stop the cluster service on that particular member, restart application services, or move an application service to another member.

The **Cluster Configuration Tool** can be displayed by choosing **Cluster => Configure** within the **Cluster Status Tool**.

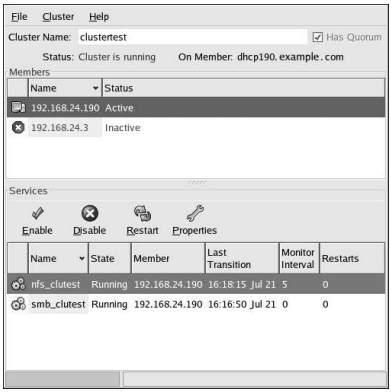


Figure 8-1. Cluster Status Tool

8.2. Displaying Cluster and Service Status

Monitoring cluster and application service status can help identify and resolve problems in the cluster environment. The following tools assist in displaying cluster status information:

- The `clustat` command
- The `clufence` command
- Log file messages
- The cluster monitoring GUI

Note that status is representative of the cluster system on which an administrator is running a particular tool. To obtain comprehensive cluster status on all member, run the tools on all cluster systems.



Important

Members that are not running the cluster software cannot determine or report the status of other members of the cluster.

Cluster and service status includes the following information:

- Cluster member system status
- Heartbeat channel status
- Service status and which cluster system is running the service or owns the service
- Service monitoring status of the cluster system

The following tables describe how to analyze the status information shown by the **Cluster Status Tool** and `clustat` command.

Member Status	Description
Active	The member system is communicating with other member systems and accessing the quorum partitions.
Inactive	The member system is unable to communicate with the other member system.

Table 8-1. Member Status

Service Status	Description
Running	The service resources are configured and available on the cluster system that owns the service.
Pending	The service has failed on a member and is pending start on another member.
Disabled	The service has been disabled, and does not have an assigned owner.
Stopped	The service not running; waiting for a member capable of starting service.
Failed	The service has failed to start on and the cluster cannot successfully stop the service. Refer to Section 4.8 *Handling Failed Services* for more information on failed services.

Table 8-2. Service Status

To display a snapshot of the current cluster status from a shell prompt, invoke the `clustat` utility.

Example output is as follows:

```
Cluster Status - clustertest                                          22:15:32
Quorum: Yes, view 36
Shared State: Shared Raw Device Driver v1.0 [Min. Size=1176064]

  Member              Status
  ------------------- ----------
  clu1                Active
  clu2                Active     <-- You are here

  Service         Status   Owner (Last)      Last Transition Chk Restarts
  --------------- -------- ----------------  --------------- --- --------
  nfs_clutest     started  clu1              22:11:28 Jul 21   0        0
  smb_clutest     started  clu2              22:13:06 Jul 21   0        0
```

To monitor the cluster and display status at specific time intervals from a shell prompt, invoke `clustat` with the `-i` *`time`* option, where *`time`* specifies the number of seconds between status snapshots. For example:

```
clustat -i 10
```

8.3. Starting and Stopping the Cluster Software

To start the cluster software on a member, run the **Cluster Status Tool** and choose **Cluster => Start Local Cluster Daemons**. The cluster member's **Status** field changes from **Disabled** to **Active**.

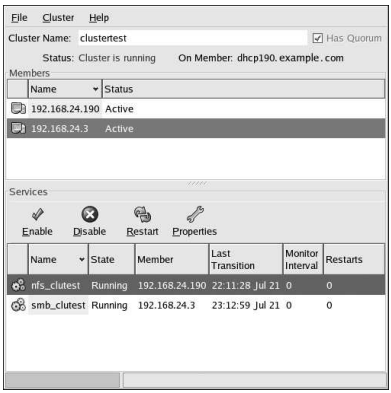


Figure 8-2. Cluster Member Started

You can also start the cluster software at a shell prompt by typing the following:

```
/sbin/service clumanager start
```

To stop a cluster member using the **Cluster Status Tool**, choose **Cluster => Stop Local Cluster Daemons**. The cluster member's **Status** field changes from **Cluster is running** to **Cluster is not running**.

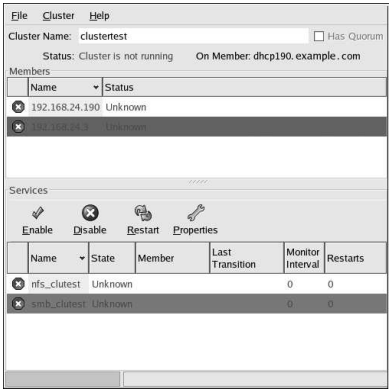


Figure 8-3. Cluster Member Stopped

To stop the cluster software on a cluster system at a shell prompt, run the following command:

```
/sbin/service clumanager stop
```

Stopping the cluster services on a member causes its services to failover to an active member.

8.4. Modifying the Cluster Configuration

It may be necessary to modify the cluster configuration. For example, it may be necessary to correct heartbeat channel or quorum partition entries in the cluster database, a copy of which is located in the `/etc/cluster.xml` file.

Important

Use only the **Cluster Configuration Tool** to modify the cluster configuration. Do *not* directly edit the `/etc/cluster.xml` file with a text editor.

To modify the cluster configuration, stop the cluster software on all cluster systems, as described in Section 8.3 *Starting and Stopping the Cluster Software*. Start the **Cluster Configuration Tool** and modify the fields that you need to change. Choose **File => Save** to save your changes. Settings automatically transfer via shared storage to active cluster members.

8.5. Backing Up and Restoring the Cluster Database

The **Cluster Configuration Tool** automatically saves a back-up copy of your `/etc/cluster.xml` file whenever you save your configuration. This is useful if the cluster does not function correctly due to misconfiguration and you need to return to a previous working configuration.

The back-up configuration is stored in a file called `/etc/cluster.xml.bak`. If a cluster member becomes inoperable due to misconfiguration, stop cluster services on all members:

```
/sbin/service cluster stop
```

Next, overwrite the erroneous configuration with a previously saved configuration:

```
cp /etc/cluster.xml.bak /etc/cluster.xml
```

Finally, re-initialize the shared storage:

```
shutil -s /etc/cluster.xml
```

8.6. Modifying Cluster Event Logging

It is possible to modify the severity level of the events that are logged by the `clupowerd`, `cluquorumd`, `cluhbd`, and `clusvcmgrd` daemons. This is done so that the daemons on the cluster systems log messages at a user-defined severity level.

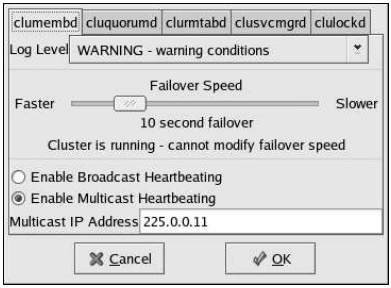


Figure 8-4. Cluster Logging Severity Levels

To change a cluster daemon's logging level on all cluster members, use the **Cluster Configuration Tool** and choose **Cluster => Daemon Properties**. Specify the daemon using the tabs, and the severity level for each daemon using the pull down menu. The following list defines each severity level:

- EMERG - The system is unusable
- ALERT - Action must be taken immediately
- CRIT - Critical conditions
- ERR - Error conditions
- WARN - Warning conditions
- NOTICE - Normal but significant condition
- INFO - Informational
- DEBUG - Debug-level messages

Note that the cluster logs messages with the designated severity level and also messages of a higher severity. For example, if the severity level for quorum daemon messages is **CRIT**, then the cluster logs messages for **CRIT**, **ALERT**, and **EMERG** severity levels. Also note that setting the logging level to a low severity level (such as **DEBUG**) results in large log files over time.

8.7. Updating the Cluster Software

To update the cluster software while minimizing service downtime, follow these steps:

1. Stop the cluster software on the first member to be upgraded by clicking **Cluster => Stop Local Cluster Daemons** from the **Cluster Status Tool**. Alternatively, run the following command at a shell prompt:
   `/sbin/service clumanager stop`
2. Make a copy of the cluster configuration file. For example, to save the file to the `/root` directory, run the following command:
   `cp /etc/cluster.xml /root/`
3. Upgrade the cluster packages using the steps outlined in Section 3.1 *Installing the Red Hat Cluster Suite Packages*.
4. Copy the configuration file you saved back to the `/etc/` directory.
   `cp /root/cluster.xml /etc/`
5. Start the cluster software on the updated cluster system by clicking **Cluster => Start Local Cluster Daemons** from the **Cluster Status Tool**. Alternatively, run the following command at a shell prompt:
   `/sbin/service clumanager start`
6. Repeat steps 1 - 5 for each cluster member to be updated.

8.8. Changing the Cluster Name

Start the **Cluster Configuration Tool** and type a new name into the **Cluster Name** field. The cluster name is used to distinctively separate multiple clusters by function or department (such as **`clu_marketing`** or **`test_cluster`**).

8.9. Disabling the Cluster Software

It may become necessary to temporarily disable the cluster software on a member system. For example, if a cluster system experiences a hardware failure, an administrator may want to reboot the system, but prevent it from rejoining the cluster to perform maintenance on the system.

Use the `/sbin/chkconfig` command to stop the member from joining the cluster at boot-up.

```
/sbin/chkconfig --del clumanager
```

Once the issues with the disabled cluster member has been resolved, use the following command to allow it to rejoin the cluster:

```
/sbin/chkconfig --add clumanager
```

You can then reboot the system for the changes to take effect or run the following command to restart cluster services:

```
/sbin/service clumanager start
```

8.10. Diagnosing and Correcting Problems in a Cluster

To ensure the proper diagnosis of any problems in a cluster, event logging must be enabled. In addition, if problems arise in a cluster, be sure to set the severity level to **DEBUG** for the cluster daemons. This logs descriptive messages that may help solve problems.



Note

Once any issues have been resolved, reset the debug level back down to its default value of **WARN** to avoid excessively large log message files from being generated. Refer to Section 8.6 *Modifying Cluster Event Logging* for more information.

Use Table 8-3 to troubleshoot issues in a cluster.

Problem	Symptom	Solution
SCSI bus not terminated	SCSI errors appear in the log file	Each SCSI bus must be terminated only at the beginning and end of the bus. Depending on the bus configuration, it might be necessary to enable or disable termination in host bus adapters, RAID controllers, and storage enclosures. To support hot plugging, external termination is required to terminate a SCSI bus. In addition, be sure that no devices are connected to a SCSI bus using a stub that is longer than 0.1 meter. Refer to Section 2.4.4 *Configuring Shared Disk Storage* and Section D.3 *SCSI Bus Termination* for information about terminating different types of SCSI buses.
SCSI bus length greater than maximum limit	SCSI errors appear in the log file	Each type of SCSI bus must adhere to restrictions on length, as described in Section D.4 *SCSI Bus Length*. In addition, ensure that no single-ended devices are connected to the LVD SCSI bus, because this causes the entire bus to revert to a single-ended bus, which has more severe length restrictions than a differential bus.
SCSI identification numbers not unique	SCSI errors appear in the log file	Each device on a SCSI bus must have a unique identification number. Refer to Section D.5 *SCSI Identification Numbers* for more information.

Problem	Symptom	Solution
SCSI commands timing out before completion	SCSI errors appear in the log file	The prioritized arbitration scheme on a SCSI bus can result in low-priority devices being locked out for some period of time. This may cause commands to time out, if a low-priority storage device, such as a disk, is unable to win arbitration and complete a command that a host has queued to it. For some workloads, this problem can be avoided by assigning low-priority SCSI identification numbers to the host bus adapters. Refer to Section D.5 *SCSI Identification Numbers* for more information.
Mounted quorum partition	Messages indicating checksum errors on a quorum partition appear in the log file	Be sure that the quorum partition raw devices are used only for cluster state information. They cannot be used for cluster services or for non-cluster purposes, and cannot contain a file system. Refer to Section 2.4.4.3 *Configuring Shared Cluster Partiti* for more information. These messages could also indicate that the underlying block device special file for the quorum partition has been erroneously used for non-cluster purposes.
Service file system is unclean	A disabled service cannot be enabled	Manually run a checking program such as `fsck`. Then, enable the service. Note that the cluster infrastructure does by default run `fsck` with the `-p` option to automatically repair file system inconsistencies. For particularly egregious error types, you may be required to manually initiate file system repair options.

Problem	Symptom	Solution
Quorum partitions not set up correctly	Messages indicating that a quorum partition cannot be accessed appear in the log file	Run the `/sbin/shutil -t` command to check that the quorum partitions are accessible. If the command succeeds, run the `shutil -p` command on both cluster systems. If the output is different on the systems, the quorum partitions do not point to the same devices on both systems. Check to make sure that the raw devices exist and are correctly specified in the `/etc/sysconfig/rawdevices` file. Refer to Section 2.4.4.3 *Configuring Shared Cluster Partitio* for more information.
Cluster service operation fails	Messages indicating the operation failed to appear on the console or in the log file	There are many different reasons for the failure of a service operation (for example, a service stop or start). To help identify the cause of the problem, set the severity level for the cluster daemons to **DEBUG** to log descriptive messages. Then, retry the operation and examine the log file. Refer to Section 8.6 *Modifying Cluster Event Logging* for more information.
Cluster service stop fails because a file system cannot be unmounted	Messages indicating the operation failed appear on the console or in the log file	Use the `fuser` and `ps` commands to identify the processes that are accessing the file system. Use the `kill` command to stop the processes. Use the `lsof -t` *`file_system`* command to display the identification numbers for the processes that are accessing the specified file system. If needed, pipe the output to the `kill` command. To avoid this problem, be sure that only cluster-related processes can access shared storage data. In addition, modify the service and enable forced unmount for the file system. This enables the cluster service to unmount a file system even if it is being accessed by an application or user.
Incorrect entry in the cluster database	Cluster operation is impaired	The **Cluster Status Tool** can be used to examine and modify service configuration. The **Cluster Configuration Tool** is used to modify cluster parameters.

Problem	Symptom	Solution
Incorrect Ethernet heartbeat entry in the cluster database or `/etc/hosts` file	Cluster status indicates that a Ethernet heartbeat channel is **OFFLINE** even though the interface is valid	Examine and modify the cluster configuration by running the **Cluster Configuration Tool**, as specified in Section 8.4 *Modifying the Cluster Configuration*, and correct the problem. In addition, be sure to use the `ping` command to send a packet to all network interfaces used in the cluster.
Loose cable connection to power switch	Power switch status using `clufence` returns an error or hangs	Check the serial cable connection.
Power switch serial port incorrectly specified in the cluster database	Power switch status using `clufence` indicates a problem	Examine the current settings and modify the cluster configuration by running the **Cluster Configuration Tool**, as specified in Section 8.4 *Modifying the Cluster Configuration*, and correct the problem.

Table 8-3. Diagnosing and Correcting Problems in a Cluster

II. Configuring a Linux Virtual Server Cluster

Building a Linux Virtual Server (LVS) system offers highly-available and scalable solution for production services using specialized routing and load-balancing techniques configured through the **Piranha Configuration Tool**. This part discusses the configuration of high-performance systems and services with Red Hat Enterprise Linux and LVS.

This section is licensed under the Open Publication License, V1.0 or later. For details refer to the Copyright page.

Table of Contents

9. Introduction to Linux Virtual Server 109
10. Linux Virtual Server Overview 111
11. Initial LVS Configuration 121
12. Setting Up a Red Hat Enterprise Linux LVS Cluster 125
13. Configuring the LVS Routers with Piranha Configuration Tool 133

Chapter 9.

Introduction to Linux Virtual Server

Using Red Hat Enterprise Linux, it is possible to create highly available server clustering solutions able to withstand many common hardware and software failures with little or no interruption of critical services. By allowing multiple computers to work together in offering these critical services, system administrators can plan and execute system maintenance and upgrades without service interruption.

The chapters in this part guide you through the following steps in understanding and deploying a clustering solution based on the Red Hat Enterprise Linux *Linux Virtual Server* (LVS) technology:

- Explains the Linux Virtual Server technology used by Red Hat Enterprise Linux to create a load-balancing cluster
- Explains how to configure a Red Hat Enterprise Linux LVS cluster
- Guides you through the **Piranha Configuration Tool**, a graphical interface used for configuring and monitoring an LVS cluster

9.1. Technology Overview

Red Hat Enterprise Linux implements highly available server solutions via clustering. It is important to note that *cluster* computing consists of three distinct branches:

- *Compute clustering* (such as Beowulf) uses multiple machines to provide greater computing power for computationally intensive tasks. This type of clustering is not addressed by Red Hat Enterprise Linux.
- *High-availability (HA) clustering* uses multiple machines to add an extra level of reliability for a service or group of services.
- *Load-balance clustering* uses specialized routing techniques to dispatch traffic to a pool of servers.

Red Hat Enterprise Linux addresses the latter two types of clustering technology. Using a collection of programs to monitor the health of the systems and services in the cluster.



Note

The clustering technology included in Red Hat Enterprise Linux is not synonymous with *fault tolerance*. Fault tolerant systems use highly specialized and often very expensive hardware to implement a fully redundant environment in which services can run uninterrupted by hardware failures.

However, fault tolerant systems do not account for operator and software errors which Red Hat Enterprise Linux can address through service redundancy. Also, since Red Hat Enterprise Linux is designed to run on commodity hardware, it creates an environment with a high level of system availability at a fraction of the cost of fault tolerant hardware.

9.2. Basic Configurations

While Red Hat Enterprise Linux can be configured in a variety of different ways, the configurations can be broken into two major categories:

- High-availability clusters using Red Hat Cluster Manager
- Load-balancing clusters using Linux Virtual Servers

This part explains what a load-balancing cluster system is and how to configure a load-balancing system using *Linux Virtual Servers* on Red Hat Enterprise Linux.

9.2.1. Load-Balancing Clusters Using Linux Virtual Servers

To an outside user accessing a hosted service (such as a website or database application), a Linux Virtual Server (LVS) cluster appears as one server. In reality, however, the user is actually accessing a cluster of two or more servers behind a pair of redundant LVS routers that distribute client requests evenly throughout the cluster system. Load-balanced clustered services allow administrators to use commodity hardware and Red Hat Enterprise Linux to create continuous and consistent access to all hosted services while also addressing availability requirements.

An LVS cluster consists of at least two layers. The first layer is composed of a pair of similarly configured Linux machines or *cluster members*. One of these machine acts as the *LVS routers*, configured to direct requests from the Internet to the cluster. The second layer consists of a cluster of machines called *real servers*. The real servers provide the critical services to the end-user while the LVS router balances the load on these servers.

For a detailed overview of LVS clustering, refer to Chapter 10 *Linux Virtual Server Overview*.

Chapter 10.

Linux Virtual Server Overview

Red Hat Enterprise Linux LVS clustering uses a Linux machine called the *active router* to send requests from the Internet to a pool of servers. To accomplish this, LVS clusters consist of two basic machine classifications — the LVS routers (one active and one backup) and a pool of real servers which provide the critical services.

The active router serves two roles in the cluster:

- To balance the load on the real servers.
- To check the integrity of the services on each of the real servers.

The backup router's job is to monitor the active router and assume its role in the event of failure.

10.1. A Basic LVS Configuration

Figure 10-1 shows a simple LVS cluster consisting of two layers. On the first layer are two LVS routers — one active and one backup. Each of the LVS routers has two network interfaces, one interface on the Internet and one on the private network, enabling them to regulate traffic between the two networks. For this example the active router is using *Network Address Translation* or *NAT* to direct traffic from the Internet to a variable number of real servers on the second layer, which in turn provide the necessary services. Therefore, the real servers in this example are connected to a dedicated private network segment and pass all public traffic back and forth through the active LVS router. To the outside world, the server cluster appears as one entity.

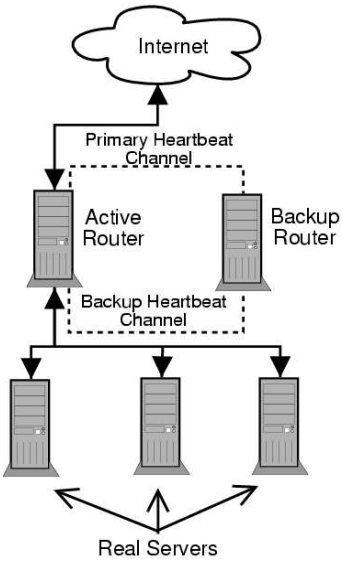


Figure 10-1. A Basic LVS Configuration

Service requests arriving at the LVS cluster are addressed to a *virtual IP* address or VIP. This is a publicly-routable address the administrator of the site associates with a fully-qualified domain name, such as www.example.com, and which is assigned to one or more *virtual server*[1]. Note that a VIP address migrates from one LVS router to the other during a failover, thus maintaining a presence at that IP address, also known as *floating IP addresses*.

VIP addresses may be aliased to the same device which connects the LVS router to the Internet. For instance, if eth0 is connected to the Internet, than multiple virtual servers can be aliased to `eth0:1`. Alternatively, each virtual server can be associated with a separate device per service. For example, HTTP traffic can be handled on `eth0:1`, and FTP traffic can be handled on `eth0:2`.

Only one LVS router is active at a time. The role of the active router is to redirect service requests from virtual IP addresses to the real servers. The redirection is based on one of eight supported load-balancing algorithms described further in Section 10.3 *LVS Scheduling Overview*.

The active router also dynamically monitors the overall health of the specific services on the real servers through simple *send/expect scripts*. To aid in detecting the health of services that require dynamic data, such as HTTPS or SSL, the administrator can also call external executables. If a service on a real server malfunctions, the active router stops sending jobs to that server until it returns to normal operation.

The backup router performs the role of a standby system. Periodically, the LVS routers exchange heartbeat messages through the primary external public interface and, in a failover situation, the private interface. Should the backup node fail to receive a heartbeat message within an expected interval, it initiates a failover and assumes the role of the active router. During failover, the backup router takes over the VIP addresses serviced by the failed router using a technique known as *ARP spoofing* — where the backup LVS router announces itself as the destination for IP packets addressed to the failed

1. A virtual server is a service configured to listen on a specific virtual IP. Refer to Section 13.6 ***VIRTUAL SERVERS*** for more on configuring a virtual server using the **Piranha Configuration Tool**.

node. When the failed node returns to active service, the backup node assumes its hot-backup role again.

The simple, two-layered configuration used in Figure 10-1 is best for clusters serving data which does not change very frequently — such as static webpages — because the individual real servers do not automatically sync data between each node.

10.1.1. Data Replication and Data Sharing Between Real Servers

Since there is no built-in component in LVS clustering to share the same data between the real servers, the administrator has two basic options:

- Synchronize the data across the real server pool
- Add a third layer to the topology for shared data access

The first option is preferred for servers that do not allow large numbers of users to upload or change data on the real servers. If the cluster allows large numbers of users to modify data, such as an e-commerce website, adding a third layer is preferable.

10.1.1.1. Configuring Real Servers to Synchronize Data

There are many ways an administrator can choose to synchronize data across the pool of real servers. For instance, shell scripts can be employed so that if a Web engineer updates a page, the page is posted to all of the servers simultaneously. Also, the cluster administrator can use programs such as `rsync` to replicate changed data across all nodes at a set interval.

However, this type of data synchronization does not optimally function if the cluster is overloaded with users constantly uploading files or issuing database transactions. For a cluster with a high load, a *three-tiered topology* is the ideal solution.

10.2. A Three Tiered LVS Configuration

Figure 10-2 shows a typical three tiered LVS cluster topology. In this example, the active LVS router routes the requests from the Internet to the pool of real servers. Each of the real servers then accesses a shared data source over the network.

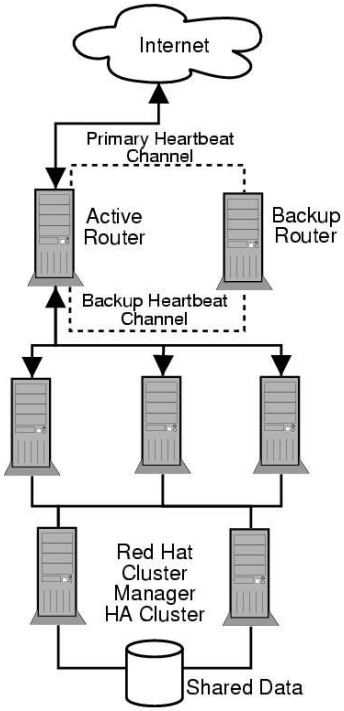


Figure 10-2. A Three Tiered LVS Configuration

This configuration is ideal for busy FTP servers, where accessible data is stored on a central, highly available server and accessed by each real server via an exported NFS directory or Samba share. This topography is also recommended for websites that access a central, highly available database for transactions. Additionally, using an active-active configuration with Red Hat Cluster Manager, administrators can configure one high-availability cluster to serve both of these roles simultaneously.

The third tier in the above example does not have to use Red Hat Cluster Manager, but failing to use a highly available solution would introduce a critical single point of failure.

10.3. LVS Scheduling Overview

One of the advantages of using an LVS cluster is its ability to perform flexible, IP-level load balancing on the real server pool. This flexibility is due to the variety of scheduling algorithms an administrator can choose from when configuring a cluster. LVS load balancing is superior to less flexible methods, such as *Round-Robin DNS* where the hierarchical nature of DNS and the caching by client machines can lead to load imbalances. Additionally, the low-level filtering employed by the LVS router has advantages over application-level request forwarding because balancing loads at the network packet level causes minimal computational overhead and allows for greater scalability.

Using scheduling, the active router can take into account the real servers' activity and, optionally, an administrator-assigned *weight* factor when routing service requests. Using assigned weights gives arbitrary priorities to individual machines. Using this form of scheduling, it is possible to create a

group of real servers using a variety of hardware and software combinations and the active router can evenly load each real server.

The scheduling mechanism for an LVS cluster is provided by a collection of kernel patches called *IP Virtual Server* or *IPVS* modules. These modules enable *layer 4* (*L4*) transport layer switching, which is designed to work well with multiple servers on a single IP address.

To track and route packets to the real servers efficiently, IPVS builds an *IPVS table* in the kernel. This table is used by the active LVS router to redirect requests from a virtual server address to and returning from real servers in the pool. The IPVS table is constantly updated by a utility called *ipvsadm* — adding and removing cluster members depending on their availability.

10.3.1. Scheduling Algorithms

The structure that the IPVS table takes depends on the scheduling algorithm that the administrator chooses for any given virtual server. To allow for maximum flexibility in the types of services you can cluster and how these services are scheduled, Red Hat Enterprise Linux provides the following scheduling algorithms listed below. For instructions on how to assign scheduling algorithms refer to Section 13.6.1 *The **VIRTUAL SERVER** Subsection*.

Round-Robin Scheduling

Distributes each request sequentially around the pool of real servers. Using this algorithm, all the real servers are treated as equals without regard to capacity or load. This scheduling model resembles round-robin DNS but is more granular due to the fact that it is network-connection based and not host-based. LVS round-robin scheduling also does not suffer the imbalances caused by cached DNS queries.

Weighted Round-Robin Scheduling

Distributes each request sequentially around the pool of real servers but gives more jobs to servers with greater capacity. Capacity is indicated by a user-assigned weight factor, which is then adjusted upward or downward by dynamic load information. Refer to Section 10.3.2 *Server Weight and Scheduling* for more on weighting real servers.

Weighted round-robin scheduling is a preferred choice if there are significant differences in the capacity of real servers in the pool. However, if the request load varies dramatically, the more heavily weighted server may answer more than its share of requests.

Least-Connection

Distributes more requests to real servers with fewer active connections. Because it keeps track of live connections to the real servers through the IPVS table, least-connection is a type of dynamic scheduling algorithm, making it a better choice if there is a high degree of variation in the request load. It is best suited for a real server pool where each member node has roughly the same capacity. If a group of servers have different capabilities, weighted least-connection scheduling is a better choice.

Weighted Least-Connections (default)

Distributes more requests to servers with fewer active connections relative to their capacities. Capacity is indicated by a user-assigned weight, which is then adjusted upward or downward by dynamic load information. The addition of weighting makes this algorithm ideal when the real server pool contains hardware of varying capacity. Refer to Section 10.3.2 *Server Weight and Scheduling* for more on weighting real servers.

Locality-Based Least-Connection Scheduling

Distributes more requests to servers with fewer active connections relative to their destination IPs. This algorithm is designed for use in a proxy-cache server cluster. It routes the packets for

an IP address to the server for that address unless that server is above its capacity and has a server in its half load, in which case it assigns the IP address to the least loaded real server.

Locality-Based Least-Connection Scheduling with Replication Scheduling

Distributes more requests to servers with fewer active connections relative to their destination IPs. This algorithm is also designed for use in a proxy-cache server cluster. It differs from Locality-Based Least-Connection Scheduling by mapping the target IP address to a subset of real server nodes. Requests are then routed to the server in this subset with the lowest number of connections. If all the nodes for the destination IP are above capacity, it replicates a new server for that destination IP address by adding the real server with the least connections from the overall pool of real servers to the subset of real servers for that destination IP. The most loaded node is then dropped from the real server subset to prevent over-replication.

Destination Hash Scheduling

Distributes requests to the pool of real servers by looking up the destination IP in a static hash table. This algorithm is designed for use in a proxy-cache server cluster.

Source Hash Scheduling

Distributes requests to the pool of real servers by looking up the source IP in a static hash table. This algorithm is designed for LVS routers with multiple firewalls.

10.3.2. Server Weight and Scheduling

The administrator of an LVS cluster can assign a *weight* to each node in the real server pool. This weight is an integer value which is factored into any *weight-aware* scheduling algorithms (such as weighted least-connections) and helps the LVS router more evenly load hardware with different capabilities.

Weights work as a ratio relative to one another. For instance, if one real server has a weight of 1 and the other server has a weight of 5, then the server with a weight of 5 gets 5 connections for every 1 connection the other server gets. The default value for a real server weight is 1.

Although adding weight to varying hardware configurations in a real server pool can help load-balance the cluster more efficiently, it can cause temporary imbalances when a real server is introduced to the real server pool and the virtual server is scheduled using weighted least-connections. For example, suppose there are three servers in the real server pool. Servers A and B are weighted at 1 and the third, server C, is weighted at 2. If server C goes down for any reason, servers A and B evenly distributes the abandoned load. However, once server C comes back online, the LVS router sees it has zero connections and floods the server with all incoming requests until it is on par with servers A and B.

To prevent this phenomenon, administrators can make the virtual server a *quiesce* server — anytime a new real server node comes online, the least-connections table is reset to zero and the LVS router routes requests as if all the real servers were newly added to the cluster.

10.4. Routing Methods

Red Hat Enterprise Linux uses *Network Address Translation* or *NAT routing* for LVS clustering, which allows the administrator tremendous flexibility when utilizing available hardware and integrating the cluster into an existing network.

10.4.1. NAT Routing

Figure 10-3, illustrates an LVS cluster utilizing NAT routing to move requests between the Internet and a private network.

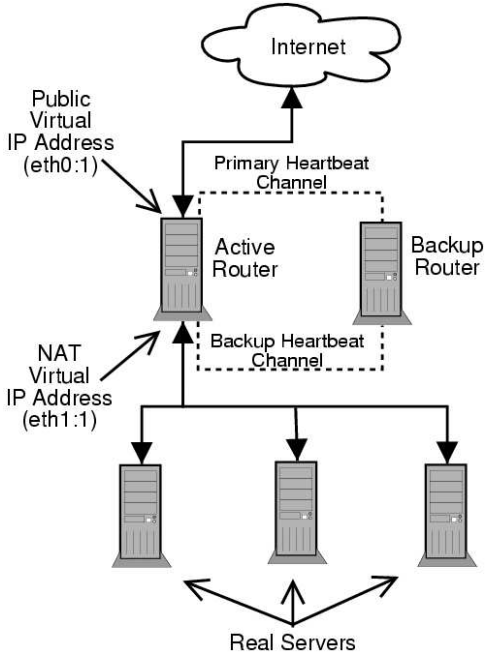


Figure 10-3. An LVS Cluster Implemented with NAT Routing

In the example, there are two NICs in the active LVS router. The NIC for the Internet has a *real IP address* on eth0 and has a floating IP address aliased to eth0:1. The NIC for the private network interface has a real IP address on eth1 and has a floating IP address aliased to eth1:1. In the event of failover, the virtual interface facing the Internet and the private facing virtual interface are taken-over by the backup LVS router simultaneously. All of the cluster's real servers located on the private network use the floating IP for the NAT router as their default route to communicate with the active LVS router so that their abilities to respond to requests from the Internet is not impaired.

In this example, the LVS router's public LVS floating IP address and private NAT floating IP address are aliased to two physical NICs. While it is possible to associate each floating IP address to its own physical device on the LVS router nodes, having more than two NICs is not a requirement.

Using this topography, the active LVS router receives the request and routes it to the appropriate server. The real server then processes the request and returns the packets to the LVS router which uses network address translation to replace the address of the real server in the packets with the LVS routers public VIP address. This process is called *IP masquerading* because the actual IP addresses of the real servers is hidden from the requesting clients.

Using this NAT routing, the real servers may be any kind of machine running various operating systems. The main disadvantage is that the LVS router may become a bottleneck in large cluster deployments because it must process outgoing as well as incoming requests.

10.5. Persistence and Firewall Marks

In certain situations, it may be desirable for a client to reconnect repeatedly to the same real server, rather than have an LVS load balancing algorithm send that request to the best available server. Examples of such situations include multi-screen web forms, cookies, SSL, and FTP connections. In these cases, a client may not work properly unless the transactions are being handled by the same server to retain context. LVS provides two different features to handle this: *persistence* and *firewall marks*.

10.5.1. Persistence

When enabled, persistence acts like a timer. When a client connects to a service, LVS remembers the last connection for a specified period of time. If that same client IP address connects again within that period, it is sent to the same server it connected to previously — bypassing the load-balancing mechanisms. When a connection occurs outside the time window, it is handled according to the scheduling rules in place.

Persistence also allows the administrator to specify a subnet mask to apply to the client IP address test as a tool for controlling what addresses have a higher level of persistence, thereby grouping connections to that subnet.

Grouping connections destined for different ports can be important for protocols which use more than one port to communicate, such as FTP. However, persistence is not the most efficient way to deal with the problem of grouping together connections destined for different ports. For these situations, it is best to use *firewall marks*.

10.5.2. Firewall Marks

Firewall marks are an easy and efficient way to a group ports used for a protocol or group of related protocols. For instance, if an LVS cluster is deployed to run an e-commerce site, firewall marks can be used to bundle HTTP connections on port 80 and secure, HTTPS connections on port 443. By assigning the same firewall mark to the virtual server for each protocol, state information for the transaction can be preserved because the LVS router forwards all requests to the same real server after a connection is opened.

Because of its efficiency and ease-of-use, administrators of LVS clusters should use firewall marks instead of persistence whenever possible for grouping connections. However, administrators should still add persistence to the virtual servers in conjunction with firewall marks to ensure the clients are reconnected to the same server for an adequate period of time.

10.6. LVS Cluster — A Block Diagram

LVS routers use a collection of programs to monitor cluster members and cluster services. Figure 10-4 illustrates how these various programs on both the active and backup LVS routers work together to manage the cluster.

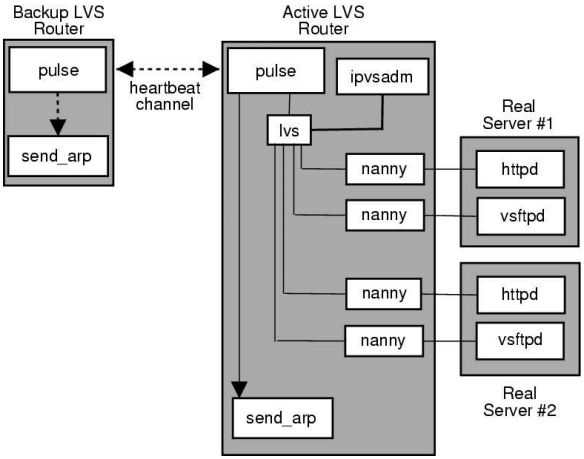


Figure 10-4. Components of a Running LVS Cluster

The `pulse` daemon runs on both the active and passive LVS routers. On the backup router, `pulse` sends a *heartbeat* to the public interface of the active router to make sure the active router is still properly functioning. On the active router, `pulse` starts the `lvs` daemon and responds to *heartbeat* queries from the backup LVS router.

Once started, the `lvs` daemon calls the `ipvsadm` utility to configure and maintain the IPVS routing table in the kernel and starts a `nanny` process for each configured virtual server on each real server. Each `nanny` process checks the state of one configured service on one real server, and tells the `lvs` daemon if the service on that real server is malfunctioning. If a malfunction is detected, the `lvs` daemon instructs `ipvsadm` to remove that real server from the IPVS routing table.

If the backup router does not receive a response from the active router, it initiates failover by calling `send_arp` to reassign all virtual IP addresses to the NIC hardware addresses (*MAC* address) of the backup node, sends a command to the active router via both the public and private network interfaces to shut down the `lvs` daemon on the active router, and starts the `lvs` daemon on the backup node to accept requests for the configured virtual servers.

10.6.1. Components of an LVS Cluster

Section 10.6.1.1 *`pulse`* shows a detailed list of each software component in an LVS router.

10.6.1.1. `pulse`

This is the controlling process which starts all other daemons related to LVS routers. At boot time, the daemon is started by the `/etc/rc.d/init.d/pulse` script. It then reads the configuration file `/etc/sysconfig/ha/lvs.cf`. On the active router, `pulse` starts the LVS daemon. On the backup router, `pulse` determines the health of the active router by executing a simple heartbeat at a user-configurable interval. If the active router fails to respond after a user-configurable interval, it initiates failover. During failover, `pulse` on the backup router instructs the `pulse` daemon on the active router to shut down all LVS services, starts the `send_arp` program to reassign the floating IP addresses to the backup router's MAC address, and starts the `lvs` daemon.

10.6.1.2. `lvs`

The `lvs` daemon runs on the active LVS router once called by `pulse`. It reads the configuration file `/etc/sysconfig/ha/lvs.cf`, calls the `ipvsadm` utility to build and maintain the IPVS routing table, and assigns a `nanny` process for each configured LVS service. If `nanny` reports a real server is down, `lvs` instructs the `ipvsadm` utility to remove the real server from the IPVS routing table.

10.6.1.3. `ipvsadm`

This service updates the IPVS routing table in the kernel. The `lvs` daemon sets up and administers an LVS cluster by calling `ipvsadm` to add, change, or delete entries in the IPVS routing table.

10.6.1.4. `nanny`

The `nanny` monitoring daemon runs on the active LVS router. Through this daemon, the active router determines the health of each real server and, optionally, monitors its workload. A separate process runs for each service defined on each real server.

10.6.1.5. `/etc/sysconfig/ha/lvs.cf`

This is the LVS cluster configuration file. Directly or indirectly, all daemons get their configuration information from this file.

10.6.1.6. Piranha Configuration Tool

This is the Web-based tool for monitoring, configuring, and administering an LVS cluster. This is the default tool to maintain the `/etc/sysconfig/ha/lvs.cf` LVS cluster configuration file.

10.6.1.7. `send_arp`

This program sends out ARP broadcasts when the floating IP address changes from one node to another during failover.

Chapter 11 *Initial LVS Configuration* reviews important post-installation configuration steps you should take before configuring Red Hat Enterprise Linux to be an LVS router.

Chapter 11.

Initial LVS Configuration

After installing Red Hat Enterprise Linux, you must take some basic steps to set up both the LVS routers and the real servers in the LVS cluster. This chapter covers these initial steps in detail.



Note

The LVS router node that becomes the active node once the cluster is started is also referred to as the *primary node*. When configuring an LVS cluster, use the **Piranha Configuration Tool** on the primary node.

11.1. Configuring Services on the LVS Routers

The Red Hat Enterprise Linux installation program installs all of the components needed to set up an LVS cluster, but the appropriate services must be activated before configuring the cluster. For both LVS routers, set the appropriate services to start at boot time. There are three primary tools available for setting services to activate at boot time under Red Hat Enterprise Linux: the command line program `chkconfig`, the ncurses-based program `ntsysv`, and the graphical **Services Configuration Tool**. All of these tools require root access.



Tip

To attain root access, open a shell prompt and type the following command followed by the root password:

```
su -
```

On the LVS routers, there are three services which need to be set to activate at boot time:

- The `piranha-gui` service (primary node only)
- The `pulse` service
- The `sshd` service

If you are clustering multi-port services or using firewall marks, you must also enable the `iptables` service.

It is best to set these services to activate in both runlevel 3 and runlevel 5. To accomplish this using `chkconfig`, type the following command for each service:

```
/sbin/chkconfig --level 35 daemon on
```

In the above command, replace *`daemon`* with the name of the service you are activating. To get a list of services on the system as well as what runlevel they are set to activate on, issue the following command:

```
/sbin/chkconfig --list
```


Warning

Turning any of the above services on using `chkconfig` does not actually start the daemon. To do this use the `/sbin/service` command. See Section 11.3 *Starting the* ***Piranha Configuration Tool*** *Service* for an example of how to use the `/sbin/service` command.

For more information on runlevels and configuring services with `ntsysv` and the **Services Configuration Tool**, refer to the chapter titled *"Controlling Access to Services"* in the *Red Hat Enterprise Linux System Administration Guide*.

11.2. Setting a Password for the Piranha Configuration Tool

Before using the **Piranha Configuration Tool** for the first time on the primary LVS router, you must restrict access to it by creating a password. To do this, login as root and issue the following command:

```
/usr/sbin/piranha-passwd
```

After entering this command, create the administrative password when prompted.


Warning

For a password to be more secure, it should not contain proper nouns, commonly used acronyms, or words in a dictionary from any language. Do not leave the password unencrypted anywhere on the system.

If the password is changed during an active **Piranha Configuration Tool** session, the administrator is prompted to provide the new password.

11.3. Starting the Piranha Configuration Tool Service

After you have set the password for the **Piranha Configuration Tool**, start or restart the `piranha-gui` service located in `/etc/rc.d/init.d/piranha-gui`. To do this, type the following command as root:

```
/sbin/service piranha-gui start
```

or

```
/sbin/service piranha-gui restart
```

Issuing this command starts a private session of the Apache HTTP Server by calling the symbolic link `/usr/sbin/piranha_gui -> /usr/sbin/httpd`. For security reasons, the `piranha-gui` version of `httpd` runs as the piranha user in a separate process. The fact that `piranha-gui` leverages the `httpd` service means that:

1. The Apache HTTP Server must be installed on the system.
2. Stopping or restarting the Apache HTTP Server via the `service` command stops the `piranha-gui` service.


Warning

If the command `/sbin/service httpd stop` or `/sbin/service httpd restart` is issued on an LVS router, you must start the `piranha-gui` service by issuing the following command:

```
/sbin/service piranha-gui start
```

The `piranha-gui` service is all that is necessary to begin configuring an LVS cluster. However, if you are configuring the cluster remotely, the `sshd` service is also required. You do *not* need to start the `pulse` service until configuration using the **Piranha Configuration Tool** is complete. See Section 13.8 *Starting the Cluster* for information on starting the `pulse` service.

11.3.1. Configuring the Piranha Configuration Tool Web Server Port

The **Piranha Configuration Tool** runs on port 3636 by default. To change this port number, change the line `Listen 3636` in Section 2 of the `piranha-gui` Web server configuration file `/etc/sysconfig/ha/conf/httpd.conf`.

To use the **Piranha Configuration Tool** you need at minimum a text-only Web browser. If you start a Web browser on the primary LVS router, open the location **`http://localhost:3636`**. You can reach the **Piranha Configuration Tool** from anywhere via Web browser by replacing *`localhost`* with the hostname or IP address of the primary LVS router.

When your browser connects to the **Piranha Configuration Tool**, you must login to access the cluster configuration services. Enter **`piranha`** in the **Username** field and the password set with `piranha-passwd` in the **Password** field.

Now that the **Piranha Configuration Tool** is running, you may wish to consider limiting who has access to the tool over the network. The next section reviews ways to accomplish this task.

11.4. Limiting Access To the Piranha Configuration Tool

The **Piranha Configuration Tool** prompts for a valid username and password combination. However, because all of the data passed to the **Piranha Configuration Tool** is in plain text, it is recommended that you restrict access only to trusted networks or to the local machine.

The easiest way to restrict access is to use the Apache HTTP Server's built in access control mechanisms by editing `/etc/sysconfig/ha/web/secure/.htaccess`. After altering the file you do not have to restart the `piranha-gui` service because the server checks the `.htaccess` file each time it accesses the directory.

By default, the access controls for this directory allow anyone to view the contents of the directory. Here is what the default access looks like:

```
Order deny,allow
Allow from all
```

To limit access of the **Piranha Configuration Tool** to only the localhost change the `.htaccess` file to allow access from only the loopback device (127.0.0.1). For more information on the loopback device, see the chapter titled *Network Scripts* in the *Red Hat Enterprise Linux Reference Guide*.

```
Order deny,allow
Deny from all
Allow from 127.0.0.1
```

You can also allow specific hosts or subnets as seen in this example:

```
Order deny,allow
Deny from all
Allow from 192.168.1.100
Allow from 172.16.57
```

In this example, only Web browsers from the machine with the IP address of 192.168.1.100 and machines on the 172.16.57/24 network can access the **Piranha Configuration Tool**.



Caution

Editing the **Piranha Configuration Tool** `.htaccess` file limits access to the configuration pages in the `/etc/sysconfig/ha/web/secure/` directory but not to the login and the help pages in `/etc/sysconfig/ha/web/`. To limit access to this directory, create a `.htaccess` file in the `/etc/sysconfig/ha/web/` directory with **`order`**, **`allow`**, and **`deny`** lines identical to `/etc/sysconfig/ha/web/secure/.htaccess`.

11.5. Turning on Packet Forwarding

In order for the LVS router to forward network packets properly to the real servers, each LVS router node must have IP forwarding turned on in the kernel. Log in as root and change the line which reads `net.ipv4.ip_forward = 0` in `/etc/sysctl.conf` to the following:

```
net.ipv4.ip_forward = 1
```

The changes take effect when you reboot the system.

To check if IP forwarding is turned on, issue the following command as root:

```
/sbin/sysctl net.ipv4.ip_forward
```

If the above command returns a `1`, then IP forwarding is enabled. If it returns a `0`, then you can turn it on manually using the following command:

```
/sbin/sysctl -w net.ipv4.ip_forward=1
```

11.6. Configuring Services on the Real Servers

If the real servers in the cluster are Red Hat Enterprise Linux systems, set the appropriate server daemons to activate at boot time. These daemons can include `httpd` for Web services or `xinetd` for FTP or Telnet services.

It may also be useful to access the real servers remotely, so the `sshd` daemon should also be installed and running.

Chapter 12.

Setting Up a Red Hat Enterprise Linux LVS Cluster

A Red Hat Enterprise Linux LVS cluster consists of two basic groups: the LVS routers and the real servers. To prevent a single point of failure, each groups should contain at least two member systems.

The LVS router group should consist of two identical or very similar systems running Red Hat Enterprise Linux. One will act as the active LVS router while the other stays in hot standby mode, so they need to have as close to the same capabilities as possible.

Before choosing and configuring the hardware for the real server group, you most decide what which of the three types of LVS topographies to use.

12.1. The NAT LVS Cluster

The NAT topography allows for great latitude in utilizing existing hardware, but it is limited in its ability to handle large loads due to the fact that all packets going into and coming out of the cluster pass through the LVS router.

Network Layout

The topography for an LVS cluster utilizing NAT routing is the easiest to configure from a network layout perspective because the cluster needs only one access point to the public network. The real servers pass all requests back through the LVS router so they are on their own private network.

Hardware

The NAT topography is the most flexible in regards to cluster hardware because the real servers do not need to be Linux machines to function correctly in the cluster. In a NAT cluster, each real server only needs one NIC since it will only be responding to the LVS router. The LVS routers, on the other hand, need two NICs each to route traffic between the two networks. Because this topography creates a network bottleneck at the LVS router, gigabit Ethernet NICs can be employed on each LVS router to increase the bandwidth the LVS routers can handle. If gigabit Ethernet is employed on the LVS routers, any switch connecting the real servers to the LVS routers must have at least two gigabit Ethernet ports to handle the load efficiently.

Software

Because the NAT topography requires the use of `iptables` for some configurations, there can be a fair amount of software configuration outside of **Piranha Configuration Tool**. In particular, FTP services and the use of firewall marks requires extra manual configuration of the LVS routers to route requests properly.

12.1.1. Configuring Network Interfaces for a NAT LVS Cluster

To set up a NAT LVS cluster, the administrator must first configure the network interfaces for the public network and the private network on the LVS routers. In this example, the LVS routers' public interfaces (`eth0`) will be on the 192.168.26/24 network (I know, I know, this is not a routable IP, but let us pretend there is a firewall in front of the LVS router for good measure) and the private interfaces which link to the real servers (`eth1`) will be on the 10.11.12/24 network.

So on the active or *primary* LVS router node, the public interface's network script, `/etc/sysconfig/network-scripts/ifcfg-eth0`, could look something like this:

```
DEVICE=eth0
BOOTPROTO=static
ONBOOT=yes
IPADDR=192.168.26.9
NETMASK=255.255.255.0
GATEWAY=192.168.26.254
```

The `/etc/sysconfig/network-scripts/ifcfg-eth1` for the private NAT interface on the LVS router could look something like this:

```
DEVICE=eth1
BOOTPROTO=static
ONBOOT=yes
IPADDR=10.11.12.9
NETMASK=255.255.255.0
```

In this example, the VIP for the LVS router's public interface will be 192.168.26.10 and the VIP for the NAT or private interface will be 10.11.12.10. So, it is essential that the real servers route requests back to the VIP for the NAT interface.



Important

The sample Ethernet interface configuration settings in this section are for the real IP addresses of an LVS router and *not* the floating IP addresses. To configure the public and private floating IP addresses the administrator should use the **Piranha Configuration Tool**, as shown in Section 13.4 ***GLOBAL SETTINGS*** and Section 13.6.1 *The **VIRTUAL SERVER** Subsection*.

After configuring the primary LVS router node's network interfaces, configure the backup LVS router's real network interfaces — taking care that none of the IP address conflict with any other IP addresses on the network.



Important

Be sure each interface on the backup node services the same network as the interface on primary node. For instance, if eth0 connects to the public network on the primary node, it must also connect to the public network on the backup node as well.

12.1.2. Routing on the Real Servers

The most important thing to remember when configuring the real servers network interfaces in a NAT cluster is to set the gateway for the NAT floating IP address of the LVS router. In this example, that address will be 10.11.12.10.



Note

Once the network interfaces are up on the real servers, the machines will be unable to ping or connect in other ways to the public network. This is normal. You will, however, be able to ping the real IP for the LVS router's private interface, in this case 10.11.12.8.

So the real server's `/etc/sysconfig/network-scripts/ifcfg-eth0` file could look similar to this:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=10.11.12.1
NETMASK=255.255.255.0
GATEWAY=10.11.12.10
```



Warning

If a real server has more than one network interface configured with a `GATEWAY=` line, the first one to come up will get the gateway. Therefore if both `eth0` and `eth1` are configured and `eth1` is used for LVS clustering, the real servers may not route requests properly.

It is best to turn off extraneous network interfaces by setting `ONBOOT=no` in their network scripts within the `/etc/sysconfig/network-scripts/` directory or by making sure the gateway is correctly set in the interface which comes up first.

12.1.3. Enabling NAT Routing on the LVS Routers

In a simple NAT LVS cluster where each clustered service uses only one port, like HTTP on port 80, the administrator needs only to enable packet forwarding on the LVS routers for the requests to be properly routed between the outside world and the real servers. See Section 11.5 *Turning on Packet Forwarding* for instructions on turning on packet forwarding. However, more configuration is necessary when the clustered services require more than one port to go to the same real server during a user session. For information on creating multi-port services using firewall marks, see Section 12.3 *Multi-port Services and LVS Clustering*.

Once forwarding is enabled on the LVS routers and the real servers are set up and have the clustered services running, use the **Piranha Configuration Tool** to configure the cluster as shown in Chapter 13 *Configuring the LVS Routers with **Piranha Configuration Tool***.



Warning

Do not configure the floating IP for `eth0:1` or `eth1:1` by manually editing network scripts or using a network configuration tool. Instead, use the **Piranha Configuration Tool** as shown in Section 13.4 ***GLOBAL SETTINGS*** and Section 13.6.1 *The **VIRTUAL SERVER** Subsection* to configure any cluster-related virtual interfaces.

When finished, start the `pulse` service as shown in Section 13.8 *Starting the Cluster*. Once `pulse` is up and running, the active LVS router will begin routing requests to the pool of real servers.

12.2. Putting the Cluster Together

After determining which of the above routing methods to use, the hardware for the LVS cluster should be linked together on the network.



Important

The adapter devices on the LVS routers must be configured to access the same networks. For instance if `eth0` connects to public network and `eth1` connects to the private network, then these same devices on the backup LVS router must connect to the same networks.

Also the gateway listed in the first interface to come up at boot time is added to the routing table and subsequent gateways listed in other interfaces are ignored. This is especially important to consider when configuring the real servers.

After physically connecting together the cluster hardware, configure the network interfaces on the primary and backup LVS routers. This can be done using a graphical application such as **redhat-config-network** or by editing the network scripts manually. For more information about adding devices using **redhat-config-network**, see the chapter titled *Network Configuration* in the *Red Hat Enterprise Linux System Administration Guide*. For more information on editing network scripts by hand, see the chapter titled *Network Scripts* in the *Red Hat Enterprise Linux Reference Guide*. For the remainder of the chapter, example alterations to network interfaces are made either manually or through the **Piranha Configuration Tool**.

12.2.1. General LVS Networking Tips

Configure the real IP addresses for both the public and private networks on the LVS routers before attempting to configure the cluster using the **Piranha Configuration Tool**. The sections on each topography give example network addresses, but the actual network addresses are needed. Below are some useful commands for bringing up network interfaces or checking their status.

Bringing Up Real Network Interfaces

The best way to bring up any real network interface is to use the following commands as root replacing *N* with the number corresponding to the interface (`eth0` and `eth1`):

```
/sbin/ifup ethN
```



Warning

Do *not* use the `ifup` scripts to bring up any floating IP addresses you may configure using **Piranha Configuration Tool** (`eth0:1` or `eth1:1`). Use the `service` command to start `pulse` instead (see Section 13.8 *Starting the Cluster* for details).

To bring a network interface down, type the following command:

```
/sbin/ifdown ethN
```

Again, replace *N* in the above command with the number corresponding to the interface you wish to bring down.

Checking the Status of Network Interfaces

If you need to check which network interfaces are up at any given time, type the following:

```
/sbin/ifconfig
```

To view the routing table for a machine, issue the following command:

```
/sbin/route
```

12.3. Multi-port Services and LVS Clustering

LVS routers under any topology require extra configuration when creating multi-port LVS services. Multi-port services can be created artificially by using firewall marks to bundle together different, but related protocols, such as HTTP (port 80) and HTTPS (port 443), or when LVS is used to cluster true multi-port protocols, such as FTP. In either case, the LVS router uses firewall marks to recognize that packets destined for different ports, but bearing the same firewall mark, should be handled identically. Also, when combined with persistence, firewall marks ensure connections from the client machine are routed to the same host, as long as the connections occur within the length of time specified by the persistence parameter. For more on assigning persistence to a virtual server, see Section 13.6.1 *The **VIRTUAL SERVER** Subsection*.

Unfortunately, the mechanism used to balance the loads on the real servers — IPVS — can recognize the firewall marks assigned to a packet, but cannot itself assign firewall marks. The job of *assigning* firewall marks must be performed by the network packet filter, `iptables`, outside of **Piranha Configuration Tool**.

12.3.1. Assigning Firewall Marks

To assign firewall marks to a packet destined for a particular port, the administrator must use `iptables`.

This section illustrates how to bundle HTTP and HTTPS as an example, however FTP is another commonly clustered multi-port protocol. If an LVS cluster is used for FTP services, see Section 12.4 *FTP In an LVS Cluster* for details on how to best configure the cluster.

The basic rule to remember when using firewall marks is that for every protocol using a firewall mark in **Piranha Configuration Tool** there must be a commensurate `iptables` rule to assign marks to the network packets.

Before creating network packet filter rules, make sure there are no rules already in place. To do this, open a shell prompt, login as root, and type:

```
/sbin/service iptables status
```

If `iptables` is not running, the prompt will instantly reappear.

If `iptables` is active, it displays a set of rules. If rules are present, type the following command:

```
/sbin/service iptables stop
```

If the rules already in place are important, check the contents of `/etc/sysconfig/iptables` and copy any rules worth keeping to a safe place before proceeding.

Below are rules which assign the same firewall mark, 80, to incoming traffic destined for the floating IP address, *`n.n.n.n`*, on ports 80 and 443. For instructions on assigning the VIP to the public network interface, see Section 13.6.1 *The **VIRTUAL SERVER** Subsection*. Also note that you must log in as root and load the module for `iptables` before issuing rules for the first time.

```
/sbin/modprobe ip_tables
/sbin/iptables -t mangle -A PREROUTING -p tcp \
     -d n.n.n.n/32 --dport 80 -j MARK --set-mark 80
/sbin/iptables -t mangle-A PREROUTING -p tcp \
               -d n.n.n.n/32 --dport 443 -j MARK --set-mark 80
```

In the above `iptables` commands, *`n.n.n.n`* should be replaced with the floating IP for your HTTP and HTTPS virtual servers. These commands have the net effect of assigning any traffic addressed to the VIP on the appropriate ports a firewall mark of 80, which in turn is recognized by IPVS and forwarded appropriately.


Warning

The commands above will take effect immediately, but do not persist through a reboot of the system. To ensure network packet filter settings are restored upon reboot, refer to Section 12.5 *Saving Network Packet Filter Settings*

12.4. FTP In an LVS Cluster

File Transport Protocol (FTP) is an old and complex multi-port protocol that presents a distinct set of challenges to a clustered environment. To understand the nature of these challenges, you must first understand some key things about how FTP works.

12.4.1. How FTP Works

With most other server client relationships, the client machine opens up a connection to the server on a particular port and the server then responds to the client on that port. When an FTP client connects to an FTP server it opens a connection to the FTP control port 21. Then the *client* tells the FTP *server* whether to establish an *active* or *passive* connection. The type of connection chosen by the client determines how the server responds and on what ports transactions will occur.

The two types of data connections are:

Active Connections

When an active connection is established, the *server* opens a data connection to the client from port 20 to a high range port on the client machine. All data from the server is then passed over this connection.

Passive Connections

When a passive connection is established, the *client* asks the FTP server to establish a passive connection port, which can be on any port higher than 10,000. The server then binds to this high-numbered port for this particular session and relays that port number back to the client. The client then opens the newly bound port for the data connection. Each data request the client makes results in a separate data connection. Most modern FTP clients attempt to establish a passive connection when requesting data from servers.

The two important things to note about all of this in regards to clustering is:

1. The *client* determines the type of connection, not the server. This means, to effectively cluster FTP, you must configure the LVS routers to handle both active and passive connections.
2. The FTP client/server relationship can potentially open a large number of ports that the **Piranha Configuration Tool** and IPVS do not know about.

12.4.2. How This Affects LVS Routing

IPVS packet forwarding only allows connections in and out of the cluster based on it recognizing its port number or its firewall mark. If a client from outside the cluster attempts to open a port IPVS is not configured to handle, it drops the connection. Similarly, if the real server attempts to open a connection back out to the Internet on a port IPVS does not know about, it drops the connection. This means *all* connections from FTP clients on the Internet *must* have the same firewall mark assigned

to them and all connections from the FTP server *must* be properly forwarded to the Internet using network packet filtering rules.

12.4.3. Creating Network Packet Filter Rules

Before assigning any `iptables` rules for FTP service, review the information in Section 12.3.1 *Assigning Firewall Marks* concerning multi-port services and techniques for checking the existing network packet filtering rules.

Below are rules which assign the same firewall mark, 21, to FTP traffic. For these rules to work properly, you must also use the **VIRTUAL SERVER** subsection of **Piranha Configuration Tool** to configure a virtual server for port 21 with a value of **`21`** in the **Firewall Mark** field. See Section 13.6.1 *The **VIRTUAL SERVER** Subsection* for details.

12.4.3.1. Rules for Active Connections

The rules for active connections tell the kernel to accept and forward connections coming to the *internal* floating IP address on port 20 — the FTP data port.

`iptables`

```
/sbin/iptables -t nat -A POSTROUTING -p tcp \
          -s n.n.n.0/24 --sport 20 -j MASQUERADE
```

In the above `iptables` commands, *`n.n.n`* should be replaced with the first three values for the floating IP for the NAT interface's internal network interface defined in the **GLOBAL SETTINGS** panel of **Piranha Configuration Tool**. The command allows the LVS router to accept outgoing connections from the real servers that IPVS does not know about.

12.4.3.2. Rules for Passive Connections

The rules for passive connections assign the appropriate firewall mark to connections coming in from the Internet to the floating IP for the service on a wide range of ports — 10,000 to 20,000.



Warning

If you are limiting the port range for passive connections, you must also configure the VSFTP server to use a matching port range. This can be accomplished by adding the following lines to `/etc/vsftpd.conf`:

```
pasv_min_port=10000
pasv_max_port=20000
```

You must also control the address that the server displays to the client for passive FTP connections. In a NAT routed LVS system, add the following line to `/etc/vsftpd.conf` to override the real server IP address to the VIP, which is what the client sees upon connection. For example:

```
pasv_address=X.X.X.X
```

Replace *`X.X.X.X`* with the VIP address of the LVS system.

For configuration of other FTP servers, consult the respective documentation.

This range should be a wide enough for most situations; however, you can increase this number to include all available non-secured ports by changing `10000:20000` in the commands below to `1024:65535`.

`iptables`

```
    /sbin/iptables -t mangle -A PREROUTING -p tcp \
                -d n.n.n.n/32  \
                --dport 21 -j MARK --set-mark 21
/sbin/iptables -t mangle -A PREROUTING -p tcp \
                -d n.n.n.n/32 \
                --dport 10000:20000 -j MARK --set-mark 21
```

In the above `iptables` commands, *`n.n.n.n`* should be replaced with the floating IP for the FTP virtual server defined in the **VIRTUAL SERVER** subsection of **Piranha Configuration Tool**. These commands have the net effect of assigning any traffic addressed to the floating IP on the appropriate ports a firewall mark of 21, which is in turn recognized by IPVS and forwarded appropriately.



Warning

The commands above take effect immediately, but do not persist through a reboot of the system. To ensure network packet filter settings are restored after a reboot, see Section 12.5 *Saving Network Packet Filter Settings*

Finally, you need to be sure that the appropriate service is set to activate on the proper runlevels. For more on this, refer to Section 11.1 *Configuring Services on the LVS Routers*.

12.5. Saving Network Packet Filter Settings

After configuring the appropriate network packet filters for your situation, save the settings so they get restored after a reboot. For `iptables`, type the following command:

```
/sbin/service iptables save
```

This saves the settings in `/etc/sysconfig/iptables` so they can be recalled at boot time.

Once this file is written, you are able to use the `/sbin/service` command to start, stop, and check the status (using the status switch) of `iptables`. The `/sbin/service` will automatically load the appropriate module for you. For an example of how to use the `/sbin/service` command, see Section 11.3 *Starting the **Piranha Configuration Tool** Service*.

Finally, you need to be sure the appropriate service is set to activate on the proper runlevels. For more on this, see Section 11.1 *Configuring Services on the LVS Routers*.

The next chapter explains how to use the **Piranha Configuration Tool** to configure the LVS router and describe the steps necessary to active an LVS cluster.

Chapter 13.

Configuring the LVS Routers with Piranha Configuration Tool

The **Piranha Configuration Tool** provides a structured approach to creating the necessary configuration file for a Piranha cluster — `/etc/sysconfig/ha/lvs.cf`. This chapter describes the basic operation of the **Piranha Configuration Tool** and how to activate the cluster once configuration is complete.



Important

The configuration file for the LVS cluster follows strict formatting rules. Using the **Piranha Configuration Tool** is the best way to prevent syntax errors in the `lvs.cf` and therefore prevent software failures.

13.1. Necessary Software

The `piranha-gui` service must be running on the primary LVS router to use the **Piranha Configuration Tool**. To configure the cluster, you minimally need a text-only Web browser, such as `links`. If you are accessing the LVS router from another machine, you also need an `ssh` connection to the primary LVS router as the root user.

While configuring the primary LVS router it is a good idea to keep a concurrent `ssh` connection in a terminal window. This connection provides a secure way to restart `pulse` and other services, configure network packet filters, and monitor `/var/log/messages` during trouble shooting.

The next four sections walk through each of the configuration pages of the **Piranha Configuration Tool** and give instructions on using it to set up the LVS cluster.

13.2. Logging Into the Piranha Configuration Tool

When configuring an LVS cluster, you should always begin by configuring the primary router with the **Piranha Configuration Tool**. To do this,verify that the `piranha-gui` service is running and an administrative password has been set, as described in Section 11.2 *Setting a Password for the **Piranha Configuration Tool***.

If you are accessing the machine locally, you can open **`http://localhost:3636`** in a Web browser to access the **Piranha Configuration Tool**. Otherwise, type in the hostname or real IP address for the server followed by **`:3636`**. Once the browser connects, you will see the screen shown in Figure 13-1.

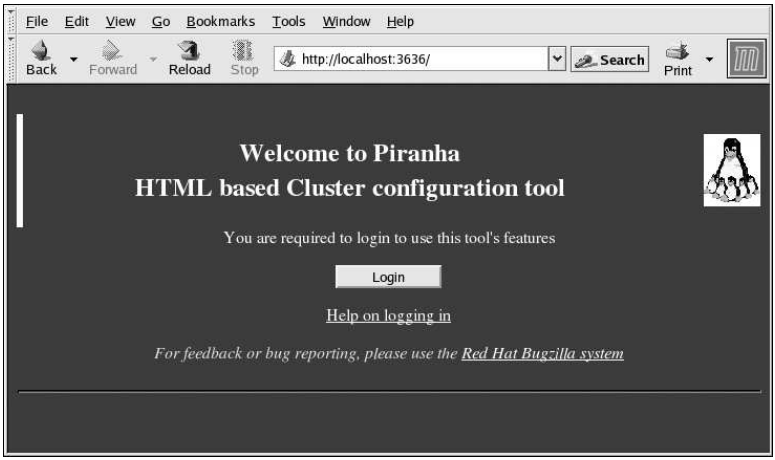


Figure 13-1. The Welcome Panel

Click on the **Login** button and enter **piranha** for the **Username** and the administrative password you created in the **Password** field.

The **Piranha Configuration Tool** is made of four main screens or *panels*. In addition, the **Virtual Servers** panel contains four *subsections*. The **CONTROL/MONITORING** panel is the first panel after the login screen.

13.3. CONTROL/MONITORING

The **CONTROL/MONITORING** Panel presents the cluster administrator with a limited runtime status of the cluster. It displays the status of the `pulse` daemon, the LVS routing table, and the LVS-spawned `nanny` processes.



Note

The fields for **CURRENT LVS ROUTING TABLE** and **CURRENT LVS PROCESSES** remain blank until you actually start the cluster, as shown in Section 13.8 *Starting the Cluster*.

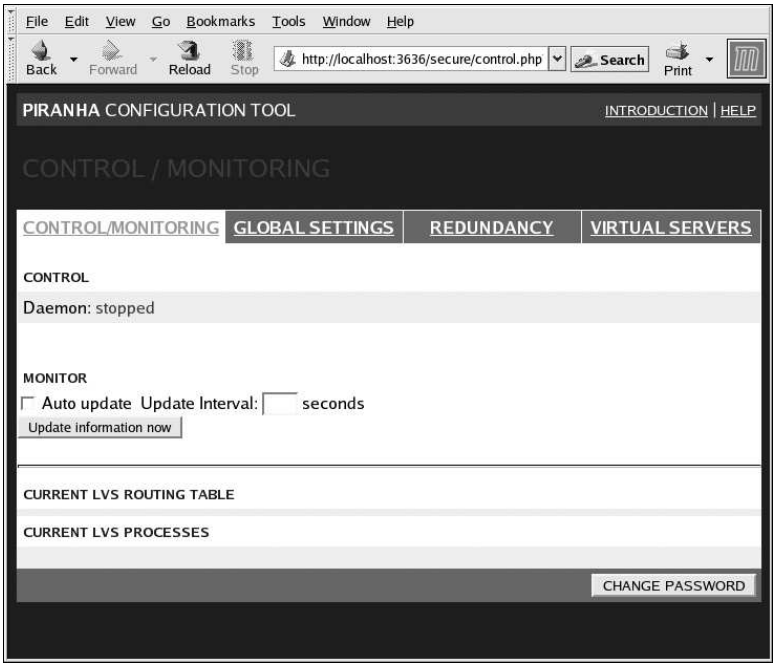


Figure 13-2. The CONTROL/MONITORING Panel

Auto update

The status display on this page can be updated automatically at a user configurable interval. To enable this feature, click on the **Auto update** checkbox and set the desired update frequency in the **Update frequency in seconds** text box (the default value is 10 seconds).

It is not recommended that you set the automatic update to an interval less than 10 seconds. Doing so may make it difficult to reconfigure the **Auto update** interval because the page will update too frequently. If you encounter this issue, simply click on another panel and then back on **CONTROL/MONITORING**.

The **Auto update** feature does not work with all browsers, such as **Mozilla**.

Update information now

You can manually update the status information manually by clicking this button.

CHANGE PASSWORD

Clicking this button takes you to a help screen with information on how to change the administrative password for the **Piranha Configuration Tool**.

13.4. GLOBAL SETTINGS

The **GLOBAL SETTINGS** panel is where the cluster administrator defines the networking details for the primary LVS router's public and private network interfaces.

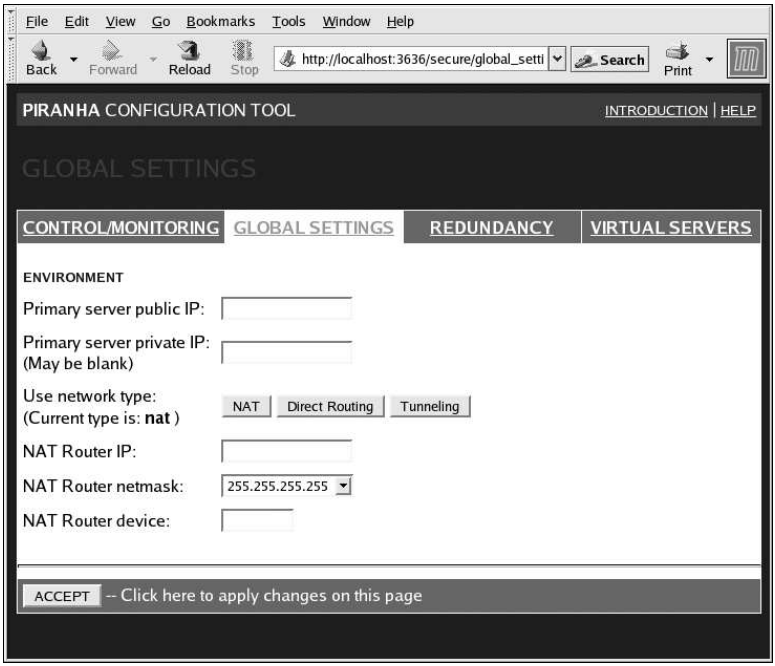


Figure 13-3. The GLOBAL SETTINGS Panel

The top half of this panel sets up the primary LVS router's public and private network interfaces. These are the interfaces already configured in Section 12.1.1 *Configuring Network Interfaces for a NAT LVS Cluster*.

Primary server public IP

In this field, enter the publicly routable real IP address for the primary LVS node.

Primary server private IP

Enter the real IP address for an alternative network interface on the primary LVS node. This address is used solely as an alternative heartbeat channel for the backup router and does not have to correlate to the real private IP address assigned in Section 12.1.1 *Configuring Network Interfaces for a NAT LVS Cluster*. You may leave this field blank, but doing so will mean there is no alternate heartbeat channel for the backup LVS router to use and therefore will create a single point of failure.



Tip

The primary LVS router's private IP can be configured on any interface that accepts TCP/IP, whether it be an Ethernet adapter or a serial port.

Use network type

Click the **NAT** button to select NAT routing.

The next three fields deal specifically with the NAT router's virtual network interface connected the private network with the real servers.

NAT Router IP

Enter the private floating IP in this text field. This floating IP should be used as the gateway for the real servers.

NAT Router netmask

If the NAT router's floating IP needs a particular netmask, select it from drop-down list.

NAT Router device

Use this text field to define the device name of the network interface for the floating IP address, such as **`eth1:1`**.



Tip

You should alias the NAT floating IP address to the Ethernet interface connected to the private network. In this example, the private network is on the `eth1` interface, so **`eth1:1`** is the floating IP address.



Warning

After completing this page, click the **ACCEPT** button to make sure you do not lose any changes when selecting a new panel.

13.5. REDUNDANCY

The **REDUNDANCY** panel allows you to configure of the backup LVS router node and set various heartbeat monitoring options.



Tip

The first time you visit this screen, it displays an "inactive" **Backup** status and an **ENABLE** button. To configure the backup LVS router, click on the **ENABLE** button so that the screen matches Figure 13-4.

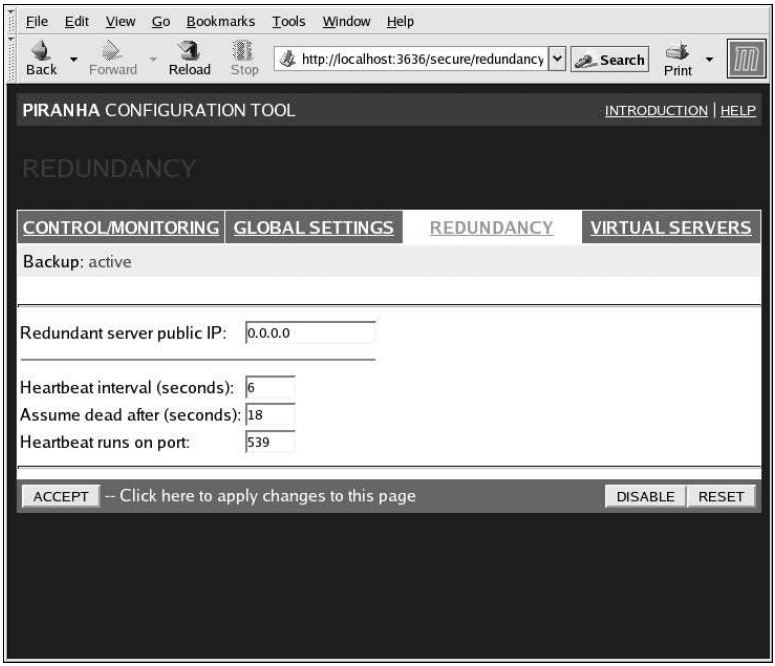


Figure 13-4. The REDUNDANCY Panel

Redundant server public IP

Enter the public real IP address for the backup LVS router node.

Redundant server private IP

Enter the backup node's private real IP address in this text field.

If you do not see the field called **Redundant server private IP**, go back to the **GLOBAL SETTINGS** panel and enter a **Primary server private IP** address and click **ACCEPT**.

The rest of the panel is devoted to configuring the heartbeat channel, which is used by the backup node to monitor the primary node for failure.

Heartbeat Interval (seconds)

This field sets the number of seconds between heartbeats — the interval that the backup node will check the functional status of the primary LVS node.

Assume dead after (seconds)

If the primary LVS node does not respond after this number of seconds, then the backup LVS router node will initiate failover.

Heartbeat runs on port

This field sets the port at which the heartbeat communicates with the primary LVS node. The default is set to 539 if this field is left blank.

Warning

Remember to click the **ACCEPT** button after making any changes in this panel to make sure you do not lose any changes when selecting a new panel.

13.6. VIRTUAL SERVERS

The **VIRTUAL SERVERS** panel displays information for each currently defined virtual server. Each table entry shows the status of the virtual server, the server name, the virtual IP assigned to the server, the netmask of the virtual IP, the port number to which the service communicates, the protocol used, and the virtual device interface.

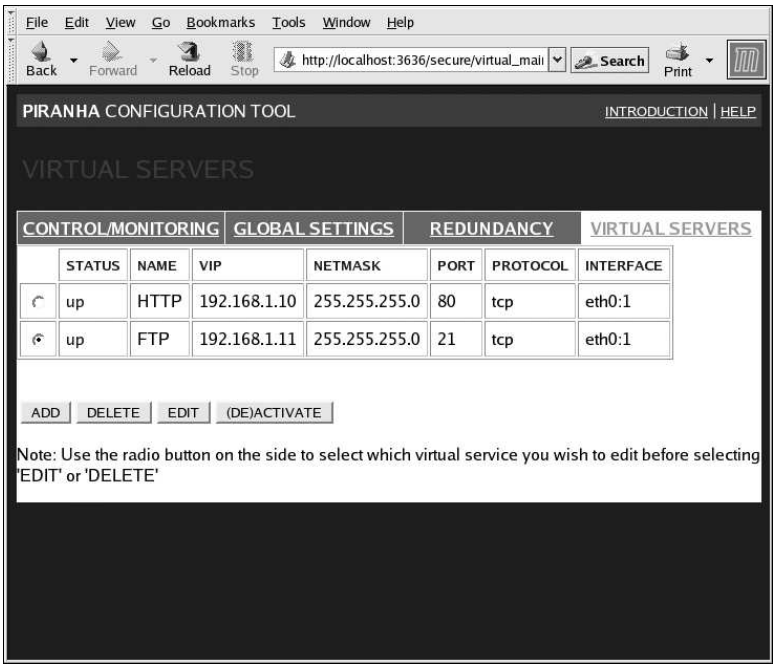


Figure 13-5. The VIRTUAL SERVERS Panel

Each server displayed in the **VIRTUAL SERVERS** panel can be configured on subsequent screens or *subsections*.

To add a service, click the **ADD** button. To remove a service, select it by clicking the radio button next to the virtual server and click the **DELETE** button.

To enable or disable a virtual server in the table click its radio button and click the **(DE)ACTIVATE** button.

After adding a virtual server, you can configure it by clicking the radio button to its left and clicking the **EDIT** button to display the **VIRTUAL SERVER** subsection.

13.6.1. The VIRTUAL SERVER Subsection

The **VIRTUAL SERVER** subsection panel shown in Figure 13-6 allows you to configure an individual virtual server. Links to subsections related specifically to this virtual server are located along the top of the page. But before configuring any of the subsections related to this virtual server, complete this page and click on the **ACCEPT** button.

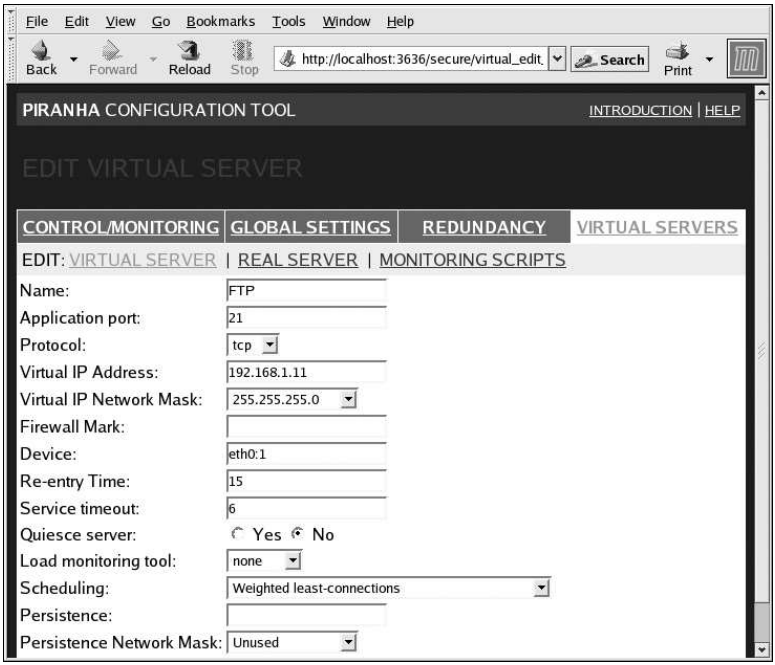


Figure 13-6. The VIRTUAL SERVERS Subsection

Name

Enter a descriptive name to identify the virtual server. This name is *not* the hostname for the machine, so make it descriptive and easily identifiable. You can even reference the protocol used by the virtual server, such as HTTP.

Application port

Enter the port number through which the service application will listen. Since this example is for HTTP services, port 80 is used.

Protocol

Choose between UDP and TCP in the drop-down menu. Web servers typically communicate via the TCP protocol, so this is selected in the example above.

Virtual IP Address

Enter the virtual server's floating IP address in this text field.

Virtual IP Network Mask

Set the netmask for this virtual server with the drop-down menu.

Firewall Mark

Do *not* enter a firewall mark integer value in this field unless you are bundling multi-port protocols or creating a multi-port virtual server for separate, but related protocols. In this example, the above virtual server has a **Firewall Mark** of 80 because we are bundling connections to HTTP on port 80 and to HTTPS on port 443 using the firewall mark value of 80. When combined with persistence, this technique will ensure users accessing both insecure and secure webpages are routed to the same real server, preserving state.



Warning

Entering a firewall mark in this field allows IPVS to recognize that packets bearing this firewall mark are treated the same, but you must perform further configuration outside of the **Piranha Configuration Tool** to actually assign the firewall marks. See Section 12.3 *Multi-port Services and LVS Clustering* for instructions on creating multi-port services and Section 12.4 *FTP In an LVS Cluster* for creating a highly available FTP virtual server.

Device

Enter the name of the network device to which you want the floating IP address defined the **Virtual IP Address** field to bind.

You should alias the public floating IP address to the Ethernet interface connected to the public network. In this example, the public network is on the `eth0` interface, so **`eth0:1`** should be entered as the device name.

Re-entry Time

Enter an integer value which defines the length of time, in seconds, before the active LVS router attempts to bring a real server back into the cluster after a failure.

Service Timeout

Enter an integer value which defines the length of time, in seconds, before a real server is considered dead and removed from the cluster.

Quiesce server

When the **Quiesce server** radio button is selected, anytime a new real server node comes online, the least-connections table is reset to zero so the active LVS router routes requests as if all the real servers were freshly added to the cluster. This option prevents the a new server from becoming bogged down with a high number of connections upon entering the cluster.

Load monitoring tool

The LVS router can monitor the load on the various real servers by using either `rup` or `ruptime`. If you select `rup` from the drop-down menu, each real server must run the `rstatd` service. If you select `ruptime`, each real server must run the `rwhod` service.



Caution

Load monitoring is *not* the same as load balancing and can result in hard to predict scheduling behavior when combined with weighted scheduling algorithms. Also, if you use load monitoring, the real servers in the cluster must be Linux machines.

Scheduling

Select your preferred scheduling algorithm from the drop-down menu. The default is **`Weighted least-connection`**. For more information on scheduling algorithms, see Section 10.3.1 *Scheduling Algorithms*.

Persistence

If an administrator needs persistent connections to the virtual server during client transactions, enter the number of seconds of inactivity allowed to lapse before a connection times out in this text field.



Important

If you entered a value in the **Firewall Mark** field above, you should enter a value for persistence as well. Also, be sure that if you use firewall marks and persistence together, that the amount of persistence is the same for each virtual server with the firewall mark. For more on persistence and firewall marks, refer to Section 10.5 *Persistence and Firewall Marks*.

Persistence Network Mask

To limit persistence to particular subnet, select the appropriate network mask from the drop-down menu.



Note

Before the advent of firewall marks, persistence limited by subnet was a crude way of bundling connections. Now, it is best to use persistence in relation to firewall marks to achieve the same result.



Warning

Remember to click the **ACCEPT** button after making any changes in this panel. To make sure you do not lose changes when selecting a new panel.

13.6.2. REAL SERVER Subsection

Clicking on the **REAL SERVER** subsection link at the top of the panel displays the **EDIT REAL SERVER** subsection. It displays the status of the physical server hosts for a particular virtual service.

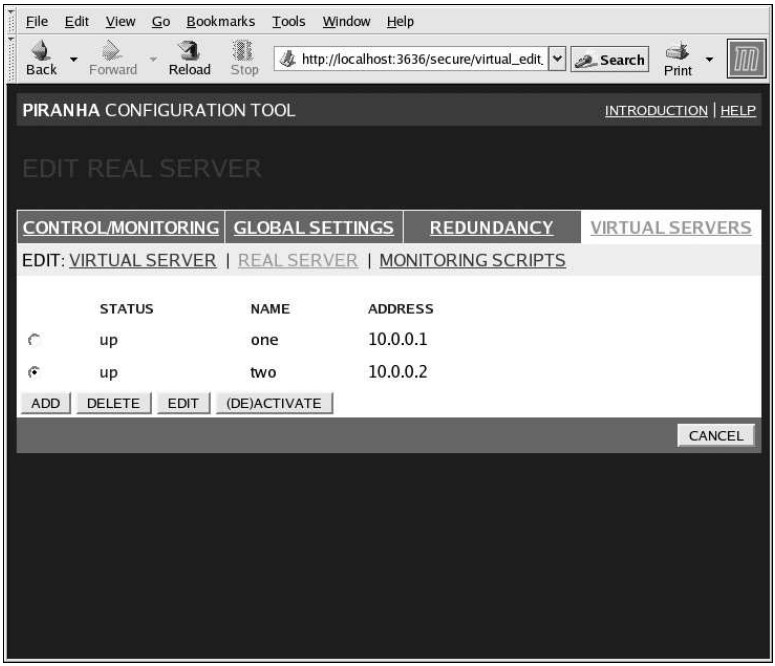


Figure 13-7. The REAL SERVER Subsection

Click the **ADD** button to add a new server. To delete an existing server, select the radio button beside it and click the **DELETE** button. Click the **EDIT** button to load the **EDIT REAL SERVER** panel, as seen in Figure 13-8.

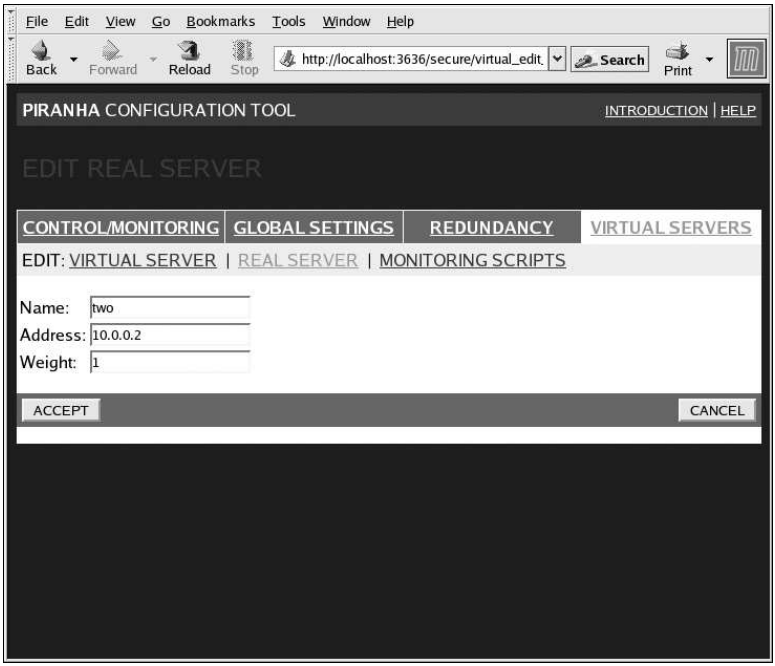


Figure 13-8. The REAL SERVER Configuration Panel

This panel consists of three entry fields:

Name

A descriptive name for the real server.



Tip

This name is *not* the hostname for the machine, so make it descriptive and easily identifiable.

Address

The real server's IP address. Since the listening port is already specified for the associated virtual server, do not add a port number.

Weight

An integer value indicating this host's capacity relative to that of other hosts in the pool. The value can be arbitrary, but treat it as a ratio in relation to other real servers in the cluster. For more on server weight, see Section 10.3.2 *Server Weight and Scheduling*.



Warning

Remember to click the **ACCEPT** button after making any changes in this panel. To make sure you do not lose any changes when selecting a new panel.

13.6.3. EDIT MONITORING SCRIPTS Subsection

Click on the **MONITORING SCRIPTS** link at the top of the page. The **EDIT MONITORING SCRIPTS** subsection allows the administrator to specify a send/expect string sequence to verify that the service for the virtual server is functional on each real server. It is also the place where the administrator can specify customized scripts to check services requiring dynamically changing data.

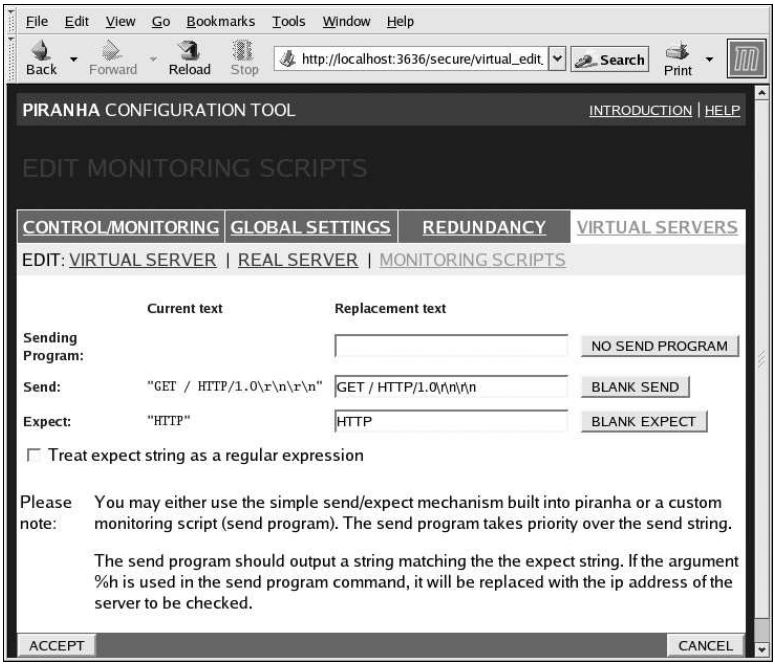


Figure 13-9. The EDIT MONITORING SCRIPTS Subsection

Sending Program

For more advanced service verification, you can use this field to specify the path to a service-checking script. This functionality is especially helpful for services that require dynamically changing data, such as HTTPS or SSL.

To use this functionality, you must write a script that returns a textual response, set it to be executable, and type the path to it in the **Sending Program** field.



Tip

To ensure that each server in the real server pool is checked, use the special token `%h` after the path to the script in the **Sending Program** field. This token is replaced with each real server's IP address as the script is called by the `nanny` daemon.

The following is a sample script to use as a guide when composing an external service-checking script:

```
#!/bin/sh

TEST=‘dig -t soa example.com @$1 | grep -c dns.example.com

if [ $TEST != "1" ]; then
 echo "OK
else
 echo "FAIL"
fi
```



Note

If an external program is entered in the **Sending Program** field, then the **Send** field is ignored.

Send

Enter a string for the `nanny` daemon to send to each real server in this field. By default the send field is completed for HTTP. You can alter this value depending on your needs. If you leave this field blank, the `nanny` daemon attempts to open the port and assume the service is running if it succeeds.

Only one send sequence is allowed in this field, and it can only contain printable, ASCII characters as well as the following escape characters:

- \n for new line.
- \r for carriage return.
- \t for tab.
- \ to escape the next character which follows it.

Expect

Enter a the textual response the server should return if it is functioning properly. If you wrote your own sending program, enter the response you told it to send if it was successful.



Tip

To determine what to send for a given service, you can open a `telnet` connection to the port on a real server and see what is returned. For instance, FTP reports 220 upon connecting, so could enter **`quit`** in the **Send** field and **`220`** in the **Expect** field.



Warning

Remember to click the **ACCEPT** button after making any changes in this panel. To make sure you do not lose any changes when selecting a new panel.

Once you have configured virtual servers using the **Piranha Configuration Tool**, you must copy specific configuration files to the backup LVS router. See Section 13.7 *Synchronizing Configuration Files* for details.

13.7. Synchronizing Configuration Files

After configuring the primary LVS router, there are several configuration files that must be copied to the backup LVS router before you start the cluster.

These files include:

- `/etc/sysconfig/ha/lvs.cf` — the configuration file for the LVS routers.
- `/etc/sysctl` — the configuration file that, among other things, turns on packet forwarding in the kernel.
- `/etc/sysconfig/iptables` — If you are using firewall marks, you should synchronize one of these files based on which network packet filter you are using.



Important

The `/etc/sysctl.conf` and `/etc/sysconfig/iptables` files do *not* change when you configure the cluster using the **Piranha Configuration Tool**.

13.7.1. Synchronizing `lvs.cf`

Anytime the LVS configuration file, `/etc/sysconfig/ha/lvs.cf`, is created or updated, you must copy it to the backup LVS router node.



Warning

Both the active and backup LVS router nodes must have identical `lvs.cf` files. Mismatched LVS configuration files between the LVS router nodes can prevent failover.

The best way to do this is to use the `scp` command.



Important

To use `scp` the `sshd` must be running on the backup router, see Section 11.1 *Configuring Services on the LVS Routers* for details on how to properly configure the necessary services on the LVS routers.

Issue the following command as the root user from the primary LVS router to sync the **`lvs.cf`** files between the router nodes:

```
scp /etc/sysconfig/ha/lvs.cf n.n.n.n:/etc/sysconfig/ha/lvs.cf
```

In the above command, replace *`n.n.n.n`* with the real IP address of the backup LVS router.

13.7.2. Synchronizing `sysctl`

The `sysctl` file is only modified once in most situations. This file is read at boot time and tells the kernel to turn on packet forwarding.



Important

If you are not sure whether or not packet forwarding is enabled in the kernel, see Section 11.5 *Turning on Packet Forwarding* for instructions on how to check and, if necessary, enable this key functionality.

13.7.3. Synchronizing Network Packet Filtering Rules

If you are using `iptables`, you will need to synchronize the appropriate configuration file on the backup LVS router.

If you alter the any network packet filter rules, enter the following command as root from the primary LVS router:

```
scp /etc/sysconfig/iptables n.n.n.n:/etc/sysconfig/
```

In the above command, replace *n.n.n.n* with the real IP address of the backup LVS router.

Next either open an `ssh` session to the backup router or log into the machine as root and type the following command:

```
/sbin/service iptables restart
```

Once you have copied these files over to the backup router and started the appropriate services (see Section 11.1 *Configuring Services on the LVS Routers* for more on this topic) you are ready to start the cluster.

13.8. Starting the Cluster

To start the LVS cluster, it is best to have two root terminals open simultaneously or two simultaneous root open `ssh` sessions to the primary LVS router.

In one terminal, watch the kernel log messages with the command:

```
tail -f /var/log/messages
```

Then start the cluster by typing the following command into the other terminal:

```
/sbin/service pulse start
```

Follow the progress of the `pulse` service's startup in the terminal with the kernel log messages. When you see the following output, the pulse daemon has started properly:

```
gratuitous lvs arps finished
```

To stop watching `/var/log/messages`, type [Ctrl]-[c].

From this point on, the primary LVS router is also the active LVS router. While you can make requests to the cluster at this point, you should start the backup LVS router before putting the cluster into service. To do this, simply repeat the process described above on the backup LVS router node.

After completing this final step, the cluster will be up and running.

III. Appendixes

This section is licensed under the GNU Free Documentation License. For details refer to the Copyright page.

Table of Contents

A. Using Red Hat Cluster Manager with Piranha 151
B. Using Red Hat GFS with Red Hat Cluster Suite 153
C. The GFS Setup Druid 157
D. Supplementary Hardware Information 163
E. Supplementary Software Information 169
F. Cluster Command-line Utilities 179

Appendix A.

Using Red Hat Cluster Manager with Piranha

A cluster can be used in conjunction with the Piranha load-balancing features to deploy a highly available e-commerce site that has complete data integrity and application availability, in addition to load balancing capabilities.

Figure A-1 shows how you could use Red Hat Cluster Manager with Piranha. The figure shows a cluster with a three-tier architecture, where the top tier consists of Piranha load-balancing systems to distribute Web requests. The second tier consists of a set of Web servers to serve the requests. The third tier consists of a cluster to serve data to the Web servers.

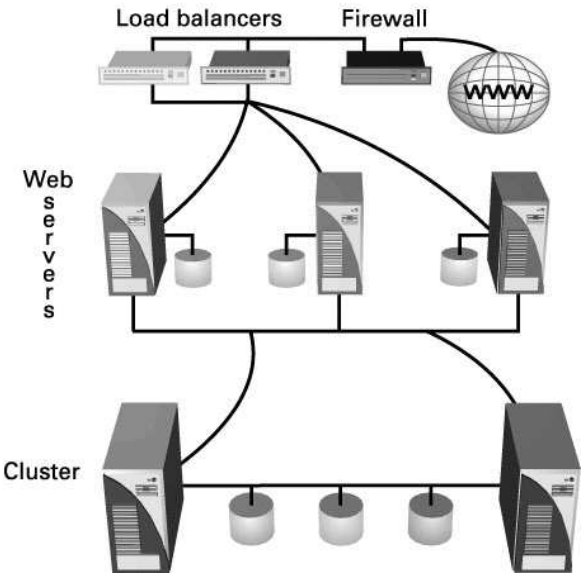


Figure A-1. Cluster in an LVS Environment

In a Piranha configuration, client systems issue requests on the World Wide Web. For security reasons, these requests enter a Web site through a firewall, which can be a Linux system serving in that capacity or a dedicated firewall device. For redundancy, you can configure firewall devices in a failover configuration. Behind the firewall are Piranha load-balancing systems, which can be configured in an active-standby mode. The active load-balancing system forwards the requests to a set of Web servers.

Each Web server can independently process an HTTP request from a client and send the response back to the client. Piranha enables an administrator to expand a Web site's capacity by adding Web servers to the load-balancing systems' set of active Web servers. In addition, if a Web server fails, it can be removed from the set.

This Piranha configuration is particularly suitable if the Web servers serve only static Web content, which consists of small amounts of infrequently changing data, such as corporate logos, that can be easily duplicated on the Web servers. However, this configuration is not suitable if the Web servers

serve dynamic content, which consists of information that changes frequently. Dynamic content could include a product inventory, purchase orders, or customer database, which must be consistent on all the Web servers to ensure that customers have access to up-to-date and accurate information.

To serve dynamic Web content in a Piranha configuration, add a cluster behind the Web servers, as shown in the previous figure. This combination of Piranha and Red Hat Cluster Manager allows for the configuration of a high-integrity, no-single-point-of-failure e-commerce site. The cluster can run a highly-available instance of a database or a set of databases that are network-accessible to the Web servers.

For example, the figure could represent an e-commerce site used for online merchandise ordering through a URL. Client requests to the URL pass through the firewall to the active Piranha load-balancing system, which then forwards the requests to one of the three Web servers. The Red Hat Cluster Manager systems serve dynamic data to the Web servers, which forward the data to the requesting client system.

Appendix B.

Using Red Hat GFS with Red Hat Cluster Suite

This appendix provides information about considerations to take when running Red Hat GFS 6.0 with Red Hat Cluster Suite and consists of the following sections:

- Section B.1 *Terminology*
- Section B.2 *Changes to Red Hat Cluster*
- Section B.3 *Installation Scenarios*

B.1. Terminology

You may have encountered new terms associated with Red Hat Cluster Suite. The following list provides a brief description of terms used with Red Hat GFS and Red Hat Cluster Suite:

GFS Setup Druid

This application is a Red Hat Cluster GUI for initial configuration of Red Hat GFS. The GUI is launched separately from the Red Hat Cluster GUI, the **Cluster Configuration Tool**. The **GFS Setup Druid** uses `/etc/cluster.xml` as input. If `/etc/cluster.xml` does not exist, the **GFS Setup Druid** displays a message and exits.



Note

You must run the **Cluster Configuration Tool** before running the **GFS Setup Druid**; the **Cluster Configuration Tool** creates `/etc/cluster.xml`.

To run the **GFS Setup Druid**, enter the following at the command line:

```
# redhat-config-gfscluster
```

gulm-bridge

This is a fence method available for Red Hat Cluster nodes, *if* and *only if* the Red Hat GFS RPM is installed on the node that the **Cluster Configuration Tool** runs on. The gulm-bridge fence method has been added to Red Hat Cluster Suite specifically for the Red Hat Enterprise Linux 4 Update 3 release. Using this fence method on a Red Hat Cluster Manager member prevents it from being fenced twice.

Red Hat Cluster

Red Hat Cluster Manager is part of the Red Hat Cluster Suite. It provides cluster administration functionality for Red Hat Enterprise Linux 4. Red Hat Cluster Manager contains two major components:

- **Red Hat Cluster Manager** — The underlying software (non-GUI) that performs Red Hat Cluster administrations services.
- **Cluster Configuration Tool** — This component is the graphical user interface (GUI) for Red Hat Cluster Manager. The GUI provides a configuration interface and a status monitor for members and services in a Red Hat Cluster Manager system. The **Cluster Configuration Tool** accepts configuration data from a user and writes it to the `/etc/cluster.xml` file. The **Red Hat Cluster Manager** reads the configuration data from the `/etc/cluster.xml` file.

Also, the **Cluster Configuration Tool** wraps several command line calls into the **Red Hat Cluster Manager**, such as starting and stopping services.

B.2. Changes to Red Hat Cluster

The following changes to Red Hat Cluster enable running it with Red Hat GFS in RHEL-U3:

- The **Cluster Configuration Tool** has been changed. After entering members in the configuration section of the application, if a member is highlighted, and you click **Add Child**, a dialog box is displayed, offering fence method options. You can select a fence method by clicking a radio button next to the fence method in the dialog box. Earlier Red Hat Cluster releases provided only two fence method options (under **Power Controller Type**): **Serial** and **Network**. For Red Hat Enterprise Linux 4 Update 3, if Red Hat GFS is installed on the node, then a third fence-method option, **GULM-STONITH** (the gulm-bridge fence method), is available.
- The **Red Hat Cluster Manager** now provides support for **GULM-STONITH**, the gulm-bridge fence method.
- A druid application, the **GFS Setup Druid**, provides for configuring an initial instance of Red Hat GFS by writing the three Red Hat GFS configuration files: `cluster.ccs`, `nodes.ccs`, and `fence.ccs`. The **GFS Setup Druid** requires an `/etc/cluster.xml` file when started.

B.3. Installation Scenarios

When running Red Hat GFS with Red Hat Cluster Manager, you must take into account certain considerations, according to the following circumstances:

- New installations of Red Hat GFS and Red Hat Cluster Manager
- Adding Red Hat GFS to an existing Red Hat Cluster Manager deployment
- Upgrading Red Hat GFS 5.2.1 to Red Hat GFS 6.0

B.3.1. New Installations of Red Hat GFS and Red Hat Cluster Manager

When installing Red Hat GFS and Red Hat Cluster Manager for the first time into a cluster, install and configure Red Hat Cluster Suite before installing and configuring Red Hat GFS. With the **Cluster Configuration Tool**, you can configure up to 16 nodes — the maximum number of nodes allowed in Red Hat Cluster Manager system.

You can add services and failover domains (and other functions) after initially configuring Red Hat GFS with the **GFS Setup Druid**.



Note

The only configuration items in Red Hat Cluster that Red Hat GFS or the **GFS Setup Druid** depend on are setting up Red Hat Cluster Manager members and specifying fence devices.

B.3.2. Adding Red Hat GFS to an Existing Red Hat Cluster Manager Deployment

Adding Red Hat GFS to an existing Red Hat Cluster Manager deployment requires running the Red Hat GFS druid application, **GFS Setup Druid** (also known as **redhat-config-gfscluster**). As with the scenario in Section B.3.1 *New Installations of Red Hat GFS and Red Hat Cluster Manager*, while Red Hat GFS is scalable up to 300 nodes, a Red Hat Cluster Manager limits the total number of nodes in a cluster to 16. Therefore, in this scenario, Red Hat GFS scalability is limited. If the 16-node limit is too small for your deployment, you may want to consider using multiple Red Hat Cluster Manager clusters.

Appendix C.

The GFS Setup Druid

The Red Hat Cluster Suite includes a graphical tool for administrators who are new to Red Hat GFS configuration. The **GFS Setup Druid** allows administrators to quickly migrate existing Red Hat Cluster Manager cluster systems to Red Hat GFS. The **GFS Setup Druid** integrates any previously configured cluster nodes and fencing devices from the existing `/etc/cluster.xml` configuration file. Administrators can then configure any new nodes and fence devices, as well as associate fence devices to nodes for proper power control.



Note

For more detailed information about installation, configuration, and administration of Red Hat GFS, refer to the *Red Hat GFS Administrator's Guide*.

The **GFS Setup Druid** can be started by typing `redhat-config-gfscluster` as root from a shell prompt.

C.1. Cluster Name

After reading the introductory screen describing the functionality of the **GFS Setup Druid**, click **Forward** to continue. You are then prompted to type a name for the GFS cluster. You can enter any name that does *not* contain spaces. (such as **`Development1`** or **`Engineering_Cluster`**). Click **Forward** to continue.

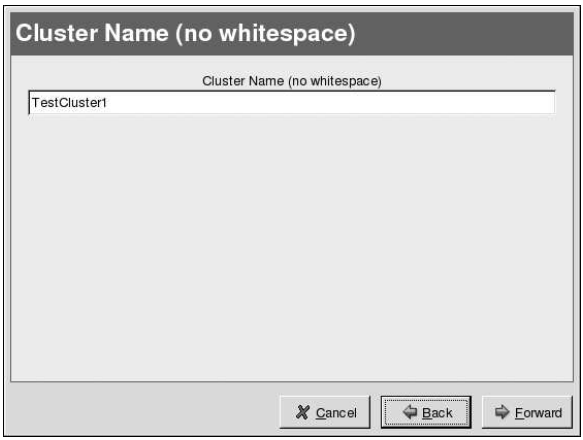


Figure C-1. Cluster Name

C.2. *LOCK_GULM* parameters

After naming the cluster, you must then configure the GFS *LOCK_GULM* parameters. The *Grand Unified Lock Manager* (GULM) is a centralized server-based locking and cluster state management for GFS file systems. Multiple lock servers can be configured for failover redundancy in the event of server failure. Figure C-2 shows the *LOCK_GULM* configuration screen.

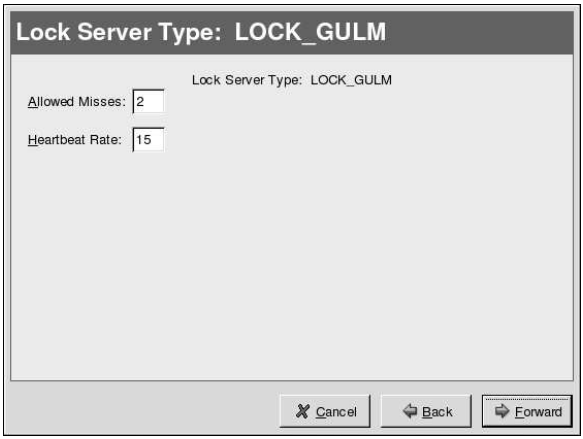


Figure C-2. Configuring LOCK_GULM parameters

Allowed Misses specifies the number of responses that a node can consecutively miss before the lock server(s) declare the node expired. The default value is 2.

Heartbeat Rate sets the interval (in seconds) at which heartbeats are checked by the lock server(s). Two-thirds of this interval is the rate at which nodes send heartbeat messages to the lock server(s). The default rate is set at 15 seconds.

After configuring the LOCK_GULM parameters, click **Forward** to continue.

C.3. Choose Location for CCS Files

The next screen prompts you to type or browse for a directory that stores the GFS cluster configuration files created by using the **GFS Setup Druid**. By default, the `/tmp/` directory stores the three configuration files: `cluster.ccs`, `nodes.ccs`, and `fence.ccs`. When a directory is chosen, click **Forward** to continue.

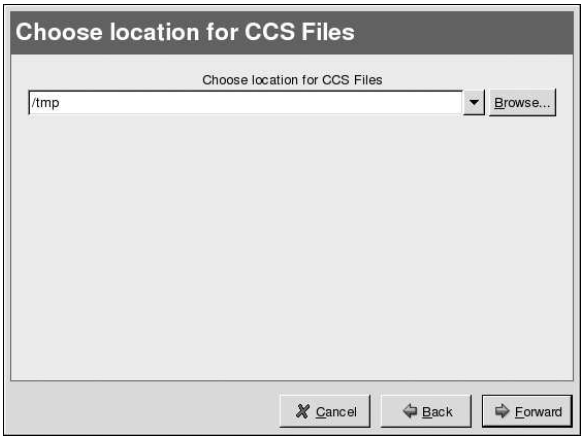


Figure C-3. Location to Write GFS Configuration Files

C.4. Cluster Members

The next screen allows you to add, delete, and configure cluster nodes. You can also configure *fence devices*, which are hardware- or software-based devices used to perform shutdown or reboot of nodes that are expired or that need to be serviced. Figure C-4 shows the cluster and fence device configuration screen.

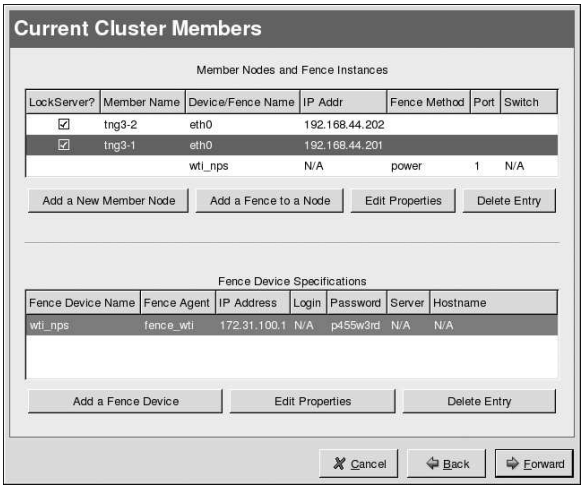


Figure C-4. Configure Cluster Members

C.4.1. Adding a New Member

Click the **Add a New Member Node** to create a new cluster member. A window similar to Figure C-5 appears that prompts you to add the **Host Name**, **IP Address**, and **Network Device** of the member node. Once these values are entered, click **Update** to add the member to the system.

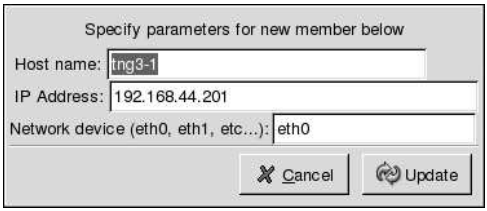


Figure C-5. Add a New Cluster Member

The added node appears within the **Member Nodes and Fence Instances** section. You can configure any member node by clicking the **Edit Properties** button. Clicking **Delete Entry** deletes the highlighted member node.

C.4.2. Fence Devices

To create an entry for a fence device, click the **Add a Fence Device** button in the **Fence Device Specifications** section. A window appears similar to Figure C-6.

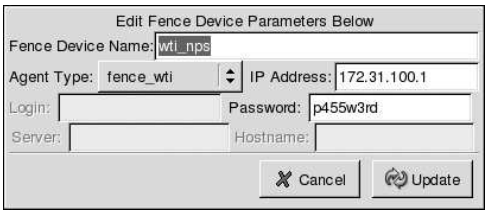


Figure C-6. Add a Fence Device

Enter a name (do not use spaces) in the **Fence Device Name** field. Then, choose the type of fence device using the **Agent Type** drop down menu. Depending on the fence device chosen, enter any required parameters. Available parameters include:

- IP Address — The address of the device if it is a network power switch or Fiber Channel switch.
- Login — The username that is entered to access the device.
- Password — The password used to authenticate the username that accesses the device.
- Server — The server name of the device (required when using the Global Network Block Device (GNBD) as a fencing method.
- Hostname — The hostname of the device. This name must be addressable to the cluster either by DNS assignment or by static entry in each nodes `/etc/hosts` file.



Note

For more information about the types of fence devices in the **Agent Type** drop down menu, refer to the chapter titled *"Creating the Cluster Configuration System Files"* in the *Red Hat GFS Administrator's Guide*.

Once a fence device is added, you can reconfigure the entry by clicking the **Edit Properties** button. To delete a device, highlight its entry and click the **Delete Entry** button.

Finally, you can associate a fence device to a cluster member by highlighting the entry for the particular member and clicking the **Add a Fence to a Node** button in the **Member Nodes and Fence Instances** section. A pop up window appears allowing you to choose any configured fence device from a drop-down menu and associate to the chosen node. Depending on the fence device chosen, you may have to enter additional fence device specifications, such as the **Port** to which the member node is connected on the fence device. Once configured, click **Update** and the **GFS Setup Druid** shows the member node with the associated fence device(s) below it.



Note

You *cannot* delete a fence device listed in the **Fence Device Specifications** section if it is associated with a cluster node. You must first delete the fence instance(s) in the **Member Nodes and Fence Instances** section for that fence device before you delete the entry in the **Fence Device Specifications** section.

C.5. Saving Your Configuration and Next Steps

When you are satisfied with your entries, click **Forward** to continue to the final screen, which prompts you to click **Apply** and save the changes to the cluster configuration files. A pop up window appears allowing you to write the files or cancel the save and go back and change the settings.



Important

Once you have written the configuration files, you *cannot* change them again using the **GFS Setup Druid**. You must either restart the configuration from the beginning or edit each cluster configuration file manually using a text editor such as `vi` or `emacs`.

Once the files are saved, you can create the cluster configuration archive (CCA), which is a binary accessible to the cluster via shared device (also called a *pool volume*).

Create the CCA by running the `ccs_tool` at a shell prompt as root. The usage of `ccs_tool` for creating the CCA is as follows:

```
ccs_tool create /directory CCADevice
```

For example, if your cluster configuration files are located in `/tmp`, and you created a shared device named `/dev/pool/devel_cca` using the GFS Pool Volume Manager (which allows you to configure pools of physical storage into logical volumes accessible to cluster nodes as if they were local storage), run the following command:

```
ccs_tool create /tmp/ /dev/pool/devel_cca
```

This chapter briefly explains the process of configuring an existing Red Hat Cluster Manager cluster to leverage Red Hat GFS using the **GFS Setup Druid**. However, for complete instructions on how to properly install, setup, run, and configure Red Hat GFS, refer to the *Red Hat GFS Administrator's Guide*.

Appendix D.

Supplementary Hardware Information

The following sections provide additional information about configuring the hardware used in a cluster system.

D.1. Setting Up Power Controllers

This section discusses power controllers. For more information about power controllers and their role in a cluster environment, refer to Section 2.1.3 *Choosing the Type of Power Controller*.

D.1.1. Power Switches

For a list of serial-attached and network-attached power switches tested with and/or supported by Red Hat, Inc. for cluster power control, refer to the *Red Hat Hardware Compatibility List* located at the following URL:

http://hardware.redhat.com/hcl/

D.1.2. Setting up Watchdog Power Switches

A description of the usage model for watchdog timers as a cluster data integrity provision appears in Section 2.1.3 *Choosing the Type of Power Controller*. As described in that section, there are two variants of watchdog timers: hardware-based and software-based.

This section details the configuration tasks required to setup watchdog timer usage in a cluster hardware configuration.

Regardless of which type of watchdog timer is employed, it is necessary to create the device special file appropriate for the watchdog timer. This can be accomplished with the following commands:

```
cd /dev
./MAKEDEV watchdog
```

When using the **Cluster Configuration Tool**, each new member added to the cluster has software watchdog functionality enabled by default.

D.1.2.1. Configuring the Software Watchdog Timer

Any cluster system can utilize the software watchdog timer as a data integrity provision, as no dedicated hardware components are required. The cluster software automatically loads the corresponding loadable kernel module called `softdog`.

If the cluster is configured to utilize the software watchdog timer, the cluster membership daemon (`clumembd`) periodically resets the timer interval. Should `clumembd` fail to reset the timer, the failed cluster member reboots itself.

When using the software watchdog timer, there is a small risk that the system hangs in such a way that the software watchdog thread is not executed. In this unlikely scenario, the other cluster member may takeover services of the apparently hung cluster member. Generally, this is a safe operation; but in the unlikely event that the hung cluster member resumes, data corruption could occur. To further lessen the chance of this vulnerability occurring when using the software watchdog timer, administrators should also configure the NMI watchdog timer as well as an external power switch (if available).

D.1.2.2. Enabling the NMI Watchdog Timer

If you are using the software watchdog timer as a data integrity provision, it is also recommended to enable the Non-Maskable Interrupt (NMI) watchdog timer to enhance the data integrity guarantees. The NMI watchdog timer is a different mechanism for causing the system to reboot in the event of a hang scenario where interrupts are blocked. This NMI watchdog can be used in conjunction with the software watchdog timer.

Unlike the software watchdog timer which is reset by the cluster quorum daemon (`cluquorumd`), the NMI watchdog timer counts system interrupts. Normally, a healthy system receives hundreds of device and timer interrupts per second. If there are no interrupts in a 5 second interval, a system hang has occurred and the NMI watchdog timer expires, initiating a system reboot.

A robust data integrity solution can be implemented by combining the health monitoring of the the cluster quorum daemon with the software watchdog timer along with the low-level system status checks of the NMI watchdog.

Correct operation of the NMI watchdog timer mechanism requires that the cluster members contain an APIC chip on the main system board.

The NMI watchdog is enabled on supported systems by adding **`nmi_watchdog=1`** to the kernel's command line. Here is an example `/etc/grub.conf`:



Note

The following GRUB and LILO bootloader configurations only apply to the x86 architecture of Red Hat Enterprise Linux.

```
# grub.conf
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title HA Test Kernel (2.4.9-10smp)
        root (hd0,0)
        # This is the kernel's command line.
        kernel /vmlinuz-2.4.9-10smp ro root=/dev/hda2 nmi_watchdog=1

# end of grub.conf
```

On systems using LILO, add **`"nmi_watchdog=1"`** to the `append` section in `/etc/lilo.conf`. For example:

```
# lilo.conf
prompt
timeout=50
default=linux
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
lba32

image=/boot/vmlinuz-2.4.9-10smp
        label=linux
        read-only
        root=/dev/hda2
        append="nmi_watchdog=1"

# end of lilo.conf
```

Run `/sbin/lilo` after editing `/etc/lilo.conf` for the changes to take effect.

To determine if a server supports the NMI watchdog timer, first try adding **`"nmi_watchdog=1"`** to the kernel command line as described above. After the system has booted, log in as root and type:

```
cat /proc/interrupts
```

The output should appear similar to the following:

```
           CPU0
  0:    5623100          XT-PIC  timer
  1:         13          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  7:          0          XT-PIC  usb-ohci
  8:          1          XT-PIC  rtc
  9:     794332          XT-PIC  aic7xxx, aic7xxx
 10:     569498          XT-PIC  eth0
 12:         24          XT-PIC  PS/2 Mouse
 14:          0          XT-PIC  ide0
NMI:    5620998
LOC:    5623358
ERR:          0
MIS:          0
```

The relevant portion of the above output is to verify that the `NMI` id appears on the left side. If `NMI` value has a value larger than zero (`0`), the server supports the NMI watchdog.

If this approach fails, that is, `NMI` is zero, try passing **`nmi_watchdog=2`** to the kernel instead of **`nmi_watchdog=1`** in the manner described previously. Again, check `/proc/interrupts` after the system boots. If `NMI` has a value larger than zero, the NMI watchdog has been configured properly. If `NMI` is zero, your system does not support the NMI watchdog timer.

D.1.2.3. Configuring a Hardware Watchdog Timer

The kernel provides driver support for various types of hardware watchdog timers. Some of these timers are implemented directly on the system board; whereas, others are separate hardware components such as PCI cards. Hardware-based watchdog timers provide excellent data integrity provisions in the cluster because they operate independently of the system processor and are therefore fully capable of rebooting a system in the event of a system hang.

Due to a lack of uniformity among low-level hardware watchdog components, it is difficult to make generalizations describing how to know if a particular system contains such components. Many low-level hardware watchdog components are not self-identifying.

When configuring any of the supported watchdog timers supported by the kernel, it is necessary to place a corresponding entry into the `/etc/modules.conf` file. For example, if an Intel-810 based TCO watchdog timer is to be used, the following line should be added to `/etc/modules.conf`:

```
alias wdt i810-tco
```

D.2. SCSI Bus Configuration Requirements

SCSI buses must adhere to a number of configuration requirements to operate correctly. Failure to adhere to these requirements adversely affects cluster operation and resource availability.

The following are SCSI bus configuration requirements:

- Buses must be terminated at each end. Refer to Section D.3 *SCSI Bus Termination* for more information.
- Buses must not extend beyond the maximum length restriction for the bus type. Internal cabling must be included in the length of the SCSI bus. Refer to Section D.4 *SCSI Bus Length* for more information.
- All devices (host bus adapters and disks) on a bus must have unique SCSI identification numbers. Refer to Section D.5 *SCSI Identification Numbers* for more information.
- The Linux device name for each shared SCSI device must be the same on each cluster system. For example, a device named `/dev/sdc` on one cluster system must be named `/dev/sdc` on the other cluster system. One way to ensure that devices are named the same is by using identical hardware for both cluster systems.

Use the system's configuration utility to set SCSI identification numbers and enable host bus adapter termination. When the system boots, a message is displayed describing how to start the utility. For example, the utility prompts the user to press [Ctrl]-[A], and follow the prompts to perform a particular task. To set storage enclosure and RAID controller termination, refer to the vendor documentation. Refer to Section D.3 *SCSI Bus Termination* and Section D.5 *SCSI Identification Numbers* for more information.

D.3. SCSI Bus Termination

A SCSI bus is an electrical path between two terminators. A device (host bus adapter, RAID controller, or disk) attaches to a SCSI bus by a short *stub*, which is an unterminated bus segment that usually must be less than 0.1 meter in length.

Buses must have only two terminators located at opposing ends of the bus. Additional terminators, terminators that are not at the ends of the bus, or long stubs cause the bus to operate incorrectly. Termination for a SCSI bus can be provided by the devices connected to the bus or by external terminators, if the internal (onboard) device termination can be disabled.

Testing has shown that external termination on HBAs that run at speeds greater than 80MB/second does not work reliably.

When disconnecting a device from a single-initiator SCSI bus follow these guidelines:

- Unterminated SCSI cables must not be connected to an operational host bus adapter or storage device.
- Connector pins must not bend or touch an electrical conductor while the SCSI cable is disconnected.
- To disconnect a host bus adapter from a single-initiator bus, first disconnect the SCSI cable from the RAID controller and then from the adapter. This ensures that the RAID controller is not exposed to any erroneous input.
- Protect connector pins from electrostatic discharge while the SCSI cable is disconnected by wearing a grounded anti-static wrist guard and physically protecting the cable ends from contact with other objects.
- Do not remove a device that is currently participating in any SCSI bus transactions.

To enable or disable an adapter's internal termination, use the system BIOS utility. When the system boots, a message is displayed describing how to start the utility. For example, many utilities prompt users to press [Ctrl]-[A]. Follow the prompts for setting the termination. At this point, it is also possible to set the SCSI identification number, as needed, and disable SCSI bus resets. Refer to Section D.5 *SCSI Identification Numbers* for more information.

To set storage enclosure and RAID controller termination, refer to the vendor documentation.

D.4. SCSI Bus Length

A SCSI bus must adhere to length restrictions for the bus type. Buses that do not adhere to these restrictions do not operate properly. The length of a SCSI bus is calculated from one terminated end to the other and must include any cabling that exists inside the system or storage enclosures.

A cluster supports LVD (low voltage differential) buses. The maximum length of a single-initiator LVD bus is 25 meters. The maximum length of a multi-initiator LVD bus is 12 meters. According to the SCSI standard, a single-initiator LVD bus is a bus that is connected to only two devices, each within 0.1 meter from a terminator. All other buses are defined as multi-initiator buses.

Do not connect any single-ended devices to an LVD bus; doing so converts the bus single-ended, which has a much shorter maximum length than a differential bus.

D.5. SCSI Identification Numbers

Each device on a SCSI bus must have a unique SCSI identification number. Devices include host bus adapters, RAID controllers, and disks.

The number of devices on a SCSI bus depends on the data path for the bus. A cluster supports wide SCSI buses, which have a 16-bit data path and support a maximum of 16 devices. Therefore, there are sixteen possible SCSI identification numbers that can be assigned to the devices on a bus.

In addition, SCSI identification numbers are prioritized. Use the following priority order to assign SCSI identification numbers:

```
7 - 6 - 5 - 4 - 3 - 2 - 1 - 0 - 15 - 14 - 13 - 12 - 11 - 10 - 9 - 8
```

The previous order specifies that 7 is the highest priority, and 8 is the lowest priority. The default SCSI identification number for a host bus adapter is 7, because adapters are usually assigned the highest priority. It is possible to assign identification numbers for logical units in a RAID subsystem by using the RAID management interface.

To modify an adapter's SCSI identification number, use the system BIOS utility. When the system boots, a message is displayed describing how to start the utility. For example, a user may be prompted to press [Ctrl]-[A] and follow the prompts for setting the SCSI identification number. At this point, it is possible to enable or disable the adapter's internal termination, as needed, and disable SCSI bus resets. Refer to Section D.3 *SCSI Bus Termination* for more information.

The prioritized arbitration scheme on a SCSI bus can result in low-priority devices being locked out for some period of time. This may cause commands to time out, if a low-priority storage device, such as a disk, is unable to win arbitration and complete a command that a host has queued to it. For some workloads, it is possible to avoid this problem by assigning low-priority SCSI identification numbers to the host bus adapters.

Appendix E.

Supplementary Software Information

The information in the following sections can assist in the management of the cluster software configuration.

E.1. Cluster Communication Mechanisms

A cluster uses several intra-cluster communication mechanisms to ensure data integrity and correct cluster behavior when a failure occurs. The cluster uses these mechanisms to:

- Control when a system can become a cluster member
- Determine the state of the cluster systems
- Control the behavior of the cluster when a failure occurs

The cluster communication mechanisms are as follows:

- Shared (quorum) partitions

  Periodically, each cluster system writes a time-stamp and system status to the primary and shadow shared partitions, which are raw partitions located on shared storage. Each member reads the system status and time-stamp that were written by the other members and determines if they are up to date. The members attempt to read the information from the primary shared partition. If this partition is corrupted, the members read the information from the shadow shared partition and simultaneously repair the primary partition. Data consistency is maintained through checksums and any inconsistencies between the partitions are automatically corrected.

  If a member reboots but cannot write to both shared partitions, the system is not allowed to join the cluster. In addition, if an existing member can no longer write to both partitions, it removes itself from the cluster by shutting down.

  Shared partitions are only used as a communication mechanism in two-member clusters that have network tie-breaker disabled.

- Remote power switch monitoring

  Periodically, each member monitors the health of the remote power switch connection, if any. The member uses this information to help determine the status of the other cluster members. The complete failure of the power switch communication mechanism does not automatically result in a failover. If a power switch fails to power-cycle a hung system, no failover is performed as the cluster infrastructure cannot guarantee the member's present state.

- Ethernet heartbeats

  The members are connected together by using point-to-point Ethernet lines. Periodically, each member issues heartbeats (pings) across these lines. The cluster uses this information to help determine the status of the members and to ensure correct cluster operation. The complete failure of the heartbeat communication mechanism does not automatically result in a failover.

If a member determines that a time-stamp from another member is not up-to-date, it checks the heartbeat status. If heartbeats to the member are still operating, the cluster software takes no action. If a member does not update its time-stamp after some period of time, and does not respond to heartbeat pings, it is considered down.

The cluster remains operational as long as one cluster system can write to the shared partitions, even if all other communication mechanisms fail.

Note that shared partition is only used as a back-up in some two-member configurations. The network membership algorithm is the primary determining factor as to which cluster members are active and which are not. A member that is not updating its time-stamp in this configuration never causes a failover unless `clumembd` reports that the member is down.

E.2. Failover and Recovery Scenarios

Understanding cluster behavior when significant events occur can assist in the proper management of a cluster. Note that cluster behavior depends on whether power switches are employed in the configuration. Power switches enable the cluster to maintain complete data integrity under all failure conditions.

The following sections describe how the system responds to various failure and error scenarios.

E.3. Common Cluster Behaviors: General

Loss of connectivity to a power switch or failure to fence a member

Common Causes: Serial power switch disconnected from controlling member. Network power switch disconnected from network.

Expected Behavior: Members controlled by the power switch will not be able to be shut down or restarted. In this case, if the member hangs, services will not fail-over from any member controlled by the switch in question.

Verification: Run `clustat` to verify that services are still marked as `running` on the member, even though it is `inactive` according to membership.

Dissolution of the cluster quorum

Common Causes: A majority of cluster members (for example, 3 of 5 members) go offline

Test Case: In a 3 member cluster, stop the cluster software on two members.

Expected Behavior: All members which do not have controlling power switches reboot immediately. All services stop immediately and their states are not updated on the shared media (when running `clustat`, the service status blocks may still display that the service is running). Service managers exit. Cluster locks are lost and become unavailable.

Verification: Run `clustat` on one of the remaining active members.

Member loses participatory status in the cluster quorum but is not hung

Common Causes: Total loss of connectivity to other members.

Test Case: Disconnect all network cables from a cluster member.

Expected Behavior: If the member has no controlling power switches, it reboots immediately. Otherwise, it attempts to stop services as quickly as possible. If a quorum exists, the set of members comprising the cluster quorum will fence the member.

`clumembd` crashes

Test Case: `killall -KILL clumembd`

Expected Behavior: System reboot.

`clumembd` hangs, watchdog in use

Test Case: `killall -STOP clumembd`

Expected Behavior: System reboot may occur if `clumembd` hangs for a time period greater than (failover_time - 1) seconds. Triggered externally by watchdog timer.

`clumembd` hangs, no watchdog in use

Test Case: `killall -STOP clumembd`

Expected Behavior: System reboot may occur if `clumembd` hangs for a time period greater than (failover_time) seconds. Triggered internally by `clumembd`.

`cluquorumd` crashes

Test Case: `killall -KILL cluquorumd`

Expected Behavior: System reboot.

`clusvcmgrd` crashes

Test Case: `killall -KILL clusvcmgrd`

Expected Behavior: `cluquorumd` re-spawns `clusvcmgrd`, which runs the stop phase of all services. Services which are stopped are started.

Verification: Consult system logs for a warning message from `cluquorumd`.

`clulockd` crashes

Test Case: `killall -KILL clulockd`

Expected Behavior: `cluquorumd` re-spawns `clulockd`. Locks may be unavailable (preventing service transitions) for a short period of time.

Verification: Consult system logs for a warning message from `cluquorumd`.

Unexpected system reboot without clean shutdown of cluster services

Common Causes: Any noted scenario which causes a system reboot.

Test Case: `reboot -fn`; pressing the reset switch.

Expected Behavior: If a power switch controls the rebooting member in question, the system will also be fenced (generally, power-cycled) if a cluster quorum exists.

Loss of quorum during clean shutdown

Test Case: Stop cluster services (`service clumanager stop`) on all members.

Expected Behavior: Any remaining services are stopped uncleanly.

Verification: Consult the system logs for warning message.

Successful STONITH fencing operation

Expected Behavior: Services on member which was fenced are started elsewhere in the cluster, if possible.

Verification: Verify that services are, in fact, started after the member is fenced. This should only take a few seconds.

Unsuccessful fencing operation on cluster member

Common Causes: Power switch returned error status or is not reachable.

Test Case: Disconnect power switch controlling a member and run `reboot -fn` on the member.

Expected Behavior: Services on a member which fails to be fenced are not started elsewhere in the cluster. If the member recovers, services on the cluster are restarted. Because there is no way to accurately determine the member's state, it is assumed that it is now still running even if heartbeats have stopped. Thus, all services should be reported as running on the down member.

Verification: Run `clustat` to verify that services are still marked as running on the member, even though it is inactive according to membership. Messages will be logged stating that the member is now in the `PANIC` state.

Error reading from one of the shared partitions

Test Case: Run `dd` to write zeros to the shared partition

```
dd if=/dev/zero of=/dev/raw/raw1 bs=512 count=1
shutil -p /cluster/header
```

Expected Behavior: Event is logged. The data from the good shared partition is copied to the partition which returned errors

Verification: A second read pass of the same data should not produce a second error message.

Error reading from both of the shared partitions

Common Causes: Shared media is either unreachable or both partitions have corruption.

Test Case: Unplug SCSI or Fibre Channel cable from a member.

Expected Behavior: The event is logged. Configured action is taken to address loss of access to shared storage (reboot/halt/stop/ignore). Default action is to reboot

E.4. Common Behaviors: Two Member Cluster with Disk-based Tie-breaker

Loss of network connectivity to other member, shared media still accessible

Common Causes: Network connectivity lost.

Test Case: Disconnect all network cables from a member.

Expected Behavior: No fail-over unless disk updates are also lost. Services will not be able to be relocated in most cases, which is due to the fact that the lock server requires network connectivity.

Verification: Run `clustat` to verify that services are still marked as running on the member, even though it is inactive according to membership. Messages are logged stating that the member is now in the `PANIC` state.

Loss of access to shared media

Common Causes: Shared media loses power, cable connecting a member to the shared media is disconnected.

Test Case: Unplug SCSI or Fibre Channel cable from a member.

Expected Behavior: No failover occurs unless network is also lost. Configured action is taken to address loss of access to shared storage (reboot/halt/stop/ignore). Default is `reboot`. The action may subsequently cause a failover.)

System hang or crash (panic) on member X

Test Case: Kill the `cluquorumd` and `clumembd` daemons.

```
killall -STOP cluquorumd clumembd
```

Expected Behavior: Hung cluster member is fenced by other cluster member. Services fail over. Configured watchdog timers may be triggered.

Start of Cluster Services without network connectivity

Common Causes: Bad switch; one or both members are without network connectivity

Test Case: Stop cluster services on all members. Disconnect all network cables from one member. Start cluster services on both members.

Verification: Not all services may start, as locks require network connectivity. *Because the Cluster Manager requires a fully connected subnet, this case is handled on a best-effort basis, but is technically an inoperable cluster.*

E.5. Common Behaviors: 2-4 Member Cluster with IP-based Tie-Breaker

Network Partition

Common Causes: Network switch problem.

Test Case: Connect half of the members to switch A. Connect other half to switch B. Connect switch A to switch B using up-link or crossover cable. Connect device acting as tie-breaker IP address to switch A. Start cluster services. Unplug switch A from switch B.

Expected Behavior: All cluster partitions which are comprised of exactly half (1/2 or 2/4) members send `ping` packets to the tie-breaker IP address. If a reply is received, the partition forms a quorum. In the test case, this means the half connected to switch A will form a quorum. *Because the Cluster Manager requires a fully connected subnet, the case where an even-split (or split-brain) occurs when both partitions can reach the tie-breaker IP is not handled.*

Verification: Run `clustat` on the members plugged in to switch A. There should be a `Cluster Quorum Incarnation` number listed at the top of the output.

Loss of access to shared media

Common Causes: Shared media loses power; cable connecting a member to the shared media is disconnected.

Test Case: Unplug SCSI or Fibre Channel cable from a member.

Expected Behavior: Configured action is taken to address loss of access to shared storage (reboot/halt/stop/ignore). Default is `reboot`.

System hang or crash (panic) on cluster member

Test Case: Kill the `cluquorumd` and `clumembd` daemons.

```
killall -STOP cluquorumd clumembd
```

Expected Behavior: The cluster member is fenced by another member. Services fail over. If a watchdog is in use, it may be triggered.

E.6. Common Behaviors: 3-5 Member Cluster

Network Partition

Common Causes: Network switch problem

Test Case: Connect a majority of members to switch A. Connect remaining members to switch B. Connect switch A to switch B using up-link or crossover cable. Start cluster services. Unplug switch A from switch B.

Expected Behavior: The partition with a majority of members continues operating, and a new view of the cluster quorum is formed. Members in the minority partition are fenced, and services which were running in the minority partition are started in the majority partition, if possible. In the test case, this means that members connected to switch A will fence members connected to switch B.

Verification: Run `clustat` on one of the members connected to switch A. There should be a `Cluster Quorum Incarnation` number listed near the top of the output.

System hang on cluster member

Test Case: Kill the `clumembd` daemon.

```
killall -STOP clumembd
```

Expected Behavior: The cluster member is fenced by another member. Services fail over. If watchdog timer is configured, it may be triggered.

Loss of access to shared media

Common Causes: Shared media loses power, cable connecting a member to the shared media is disconnected.

Test Case: Unplug SCSI or Fibre Channel cable from a member.

Expected Behavior: Configured action is taken to address loss of access to shared storage (reboot/halt/stop/ignore). Default is `reboot`.

E.7. Common Behaviors: Cluster Service Daemons

Service status check fails

Common Causes: User script reports error, `clurmtabd` not running on NFS service, `smbd` and `nmbd` not running for a service with a Samba share.

Test Case: Create a service with an init script that returns a `status` output of `1`.

Expected Behavior: Service restarts on the current owner.

Verification: Consult system logs for a service restart event. The `restarts` field of the service's status information should be incremented.

A member fails to start services

Common Causes: User script returns error due to file system errors.

Test Case: Create a service with a user script which returns 1 for the `start` phase only on one member. Attempt to enable the service on this member.

Expected Behavior: Service is stopped and started on another member, provided the services stops successfully.

Service start fails on all members

Common Causes: User script returns error, file system errors.

Test Case: Create a service with a user script which returns 1 for the `start` phase on all members.

Expected Behavior: Service is placed into the `disabled` state.

Verification: Run `clustat` and verify that the service is in the `disabled state`.

Service stop fails on a member

Common Causes: User script returns error; file system can not be unmounted.

Test Case: Create a service script which returns 1 for the `stop` phase.

Expected Behavior: Service is placed into the `failed` state. At this point, the administrator must intervene to determine the cause of the failure and the appropriate course of action. The service must then be disabled before it can be enabled.

Verification: Run `clustat` and verify that the service has been placed in to the `failed` state.

E.8. Common Behaviors: Miscellaneous

`/etc/cluster.xml` no longer exists yet cluster is running

Test Case: `rm -f cluster.xml`

Expected Behavior: `clusvcmgrd` recreates `/etc/cluster.xml` from its cached version.

Verification: Check to see that `/etc/cluster.xml` is present shortly after testing this action. A log message appears in the system logs.

`clurmtabd` crashes

Test Case: Kill the `clurmtabd` daemon.
`killall -KILL clurmtabd`

Expected Behavior: Since `clurmtabd` is only spawned during the start phase of a service with NFS exports, it is only checked by a service which has a **Check Interval** and NFS exports configured. If `clurmtabd` is not present (and it is configured to be), the check phase of the service returns an error and the service is restarted.

Verification: Consult system logs for a service restart event. The `restarts` field of the service's status information should be incremented.

E.9. The `cluster.xml` File

The cluster configuration file, `/etc/cluster.xml`, contains detailed information about the cluster members and services. *Do not* manually edit the configuration file. Instead, use the **Cluster Configuration Tool** to modify the cluster configuration.

When you run **Cluster Configuration Tool**, cluster-specific information is entered in a hierarchical XML format. The following is a description of each area of configuration, from daemon and shared storage to cluster members and services. Note that the back slash (\) represents a continuation of a single line.

```
<?xml version="1.0"?>
<cluconfig version="3.0">
  <clumembd broadcast="no" interval="500000" loglevel="4" multicast="yes" \
     multicast_ipaddress="225.0.0.11" thread="yes" tko_count="20"/>
  <cluquorumd loglevel="6" pinginterval="" tiebreaker_ip=""/>
  <clurmtabd loglevel="4" pollinterval="4"/>
  <clusvcmgrd loglevel="4"/>
  <clulockd loglevel="4"/>
  <cluster config_viewnumber="18" key="7a497d303feefeef0f8be9b72697aaed" name="Octane"/>
```

The above fields contain versioning information and cluster daemon operation parameters such as logging levels, networking addresses, and more. For more information on configuring cluster daemon parameters, refer to Section 3.6 *Configuring Cluster Daemons*.

```
<cluster config_viewnumber="18" key="7a497d303feefeef0f8be9b72697aaed" \
   name="Test_cluster"/>
<sharedstate driver="libsharedraw.so" rawprimary="/dev/raw/raw1" \
   rawshadow="/dev/raw/raw2" type="raw"/>
```

The fields above define the cluster quorum and shared cluster configuration parameters. The driver and raw partition information about the shared primary and backup partitions are also specified in these fields. For more information about configuring the shared partitions, refer to Section 3.5 *Editing the* `rawdevices` *File*.

```
  <members>
    <member id="0" name="clu1" watchdog="yes">
      <powercontroller id="0" ipaddress="192.168.65.51" password="apc" \
   port="1:1" type="apcmaster" user="apc"/>
    </member>
    <member id="1" name="clu2" watchdog="yes">
      <powercontroller id="0" ipaddress="192.168.65.52" password="baytech" \
   port="1" type="baytech" user="admin"/>
    </member>
    <member id="2" name="clu3" watchdog="yes">
      <powercontroller id="0" ipaddress="192.168.65.53" password="baytech" \
   port="2" type="baytech" user="admin"/>
    </member>
    <member id="3" name="clu4" watchdog="yes">
      <powercontroller id="0" ipaddress="192.168.65.54" password="wti" \
   port="blue" type="wti_nps" user=""/>
    </member>
    </members>
```

The fields above define the cluster and its individual members. Each member field contains identification and configuration information, including cluster names, addresses, power controllers and types, and authentication details. For more information on configuring cluster members, refer to Section 3.7 *Adding and Deleting Members*.

```
  <services>
    <service checkinterval="0" failoverdomain="None" id="0" name="test" \
 userscript="None">
      <service_ipaddresses/>
    </service>
    <service checkinterval="0" failoverdomain="foodomain" id="1" name="test2" \
 userscript="None">
      <service_ipaddresses/>
    </service>
  </services>
```

The fields above define the services controlled by the cluster system, such as NFS, Samba, and HTTP. The parameters in these fields include service names, failover domain names, service status check intervals, and location of service init scripts (if applicable). For more information about configuring clustered services, refer to Section 3.10 *Adding a Service to the Cluster*.

```
  <failoverdomains>
    <failoverdomain id="0" name="fonfs" ordered="yes" restricted="yes">
      <failoverdomainnode id="0" name="clu2"/>
      <failoverdomainnode id="1" name="clu3"/>
    </failoverdomain>
    <failoverdomain id="1" name="fosamba" ordered="no" restricted="no">
      <failoverdomainnode id="0" name="clu1"/>
      <failoverdomainnode id="1" name="clu3"/>
    </failoverdomain>
  </failoverdomains>
```

The fields above define the failover domains that control the priority and order in which the cluster members queue in the event of failover. Parameters in these fields include failover domain name, restricted and ordered toggling, and node order by member name. For more information about configuring failover domains for cluster systems, refer to Section 3.9 *Configuring a Failover Domain*.

Appendix F.

Cluster Command-line Utilities

This appendix provides reference information on the following command-line utilities provided with Red Hat Cluster Suite:

- `redhat-config-cluster-cmd`—Provides command-line access to the configuration features of the **Cluster Configuration Tool** utility
- `shutil`—Checks status of the quorum partitions
- `clufence`—Tests and controls the connections to network and serial-attached power switches

F.1. Using `redhat-config-cluster-cmd`

This section details an example of the **redhat-config-cluster-cmd** utility, which allows you to configure all aspects of the cluster, and stores the information in the `/etc/cluster.xml` file.

Usage, options, and examples of using the `redhat-config-cluster-cmd` command can be found in its man page. To access the manpage from a shell-prompt, type `man redhat-config-cluster-cmd`.

The following describes an example cluster system that is configured using only the `redhat-config-cluster-cmd` utility.

Suppose a system administrator wants to create a cluster system that will serve highly available NFS services to the engineering department of a small organization. The NFS export should only be accessible to the three members of the department (Bob, Jane, and Tom).

1. Add the service and assign it a descriptive name to distinguish its functionality from other services that may run on the cluster.
   ```
   redhat-config-cluster-cmd --add_service --name=nfs_engineers
   ```
2. Add a service IP address that will transfer from one member to another in the event of failover:
   ```
   redhat-config-cluster-cmd --service=nfs_engineers --add_service_ipaddress \
          --ipaddress=10.0.0.10
   ```
3. Add a device to the service (the disk partition that serves as the NFS export):
   ```
   redhat-config-cluster-cmd --service=nfs_engineering --add_device --name=/dev/sdc3
   ```
4. Add a mount point for the device (note: the mount point cannot be listed in `/etc/fstab`):
   ```
   redhat-config-cluster-cmd --service=nfs_engineering --device=/dev/sdc3 --mount \
          --mountpoint=/mnt/nfs/engineering/ --fstype=ext3 \
          --options=rw,nosuid,sync --forceunmount=yes
   ```
5. Add the mounted directory for NFS exporting:
   ```
   redhat-config-cluster-cmd --service=nfs_engineering --device=/dev/sdc3 \
          --add_nfs_export --name=/mnt/nfs/engineering
   ```
6. Allow Bob to access the clustered NFS export:
   ```
   redhat-config-cluster-cmd --service=nfs_engineering --device=/dev/sdc3 \
          --nfsexport=/mnt/nfs/engineering --add_client --name=bob \
          --options=rw
   ```
7. Repeat step 6 for Jane and Tom.

For more information and examples of using `redhat-config-cluster-cmd`, refer to the man page by typing the following at a shell prompt:

```
man redhat-config-cluster-cmd
```

F.2. Using the `shutil` Utility

Test the quorum partitions and ensure that they are accessible by invoking the `shutil` utility with the `-p` option.

Running `/usr/sbin/shutil -p /cluster/header` on all active cluster members should give the same output on each machine. For example:

```
/cluster/header is 144 bytes long
SharedStateHeader {
 ss_magic = 0x39119fcd
 ss_timestamp = 0x000000003f5d3eea (22:46:02 Sep 08 2003)
 ss_updateHost = lab.example.com
}
```

If the output of the `shutil` utility with the `-p` option is not the same on all cluster systems, perform the following:

- Examine the `/etc/sysconfig/rawdevices` file on each cluster system and ensure that the raw character devices and block devices for the primary and backup quorum partitions have been accurately specified. If they are not the same, edit the file and correct any mistakes. Then re-run the **cluconfig** utility. Refer to Section 3.5 *Editing the `rawdevices` File* for more information.
- Ensure that you have created the raw devices for the quorum partitions on each cluster system. Refer to Section 2.4.4.3 *Configuring Shared Cluster Partitions* for more information.
- On each cluster system, examine the system startup messages at the point where the system probes the SCSI subsystem to determine the bus configuration. Verify that both cluster systems identify the same shared storage devices and assign them the same name.
- Verify that a cluster system is not attempting to mount a file system on the quorum partition. For example, make sure that the actual device (for example, `/dev/sdb1`) is not included in an `/etc/fstab` file.

F.3. Using the `clusvcadm` Utility

The `clusvcadm` utility provides a command-line user interface that enables an administrator to monitor and manage the cluster systems and services. Use the `clusvcadm` utility to perform the following tasks:

- Disable and enable services
- Relocate and restart cluster services
- lock and unlock service state

The `clusvcadm` command line options are as follows:

`-d` *`service`*

Disable a service.

`-e` *`service`*

Enable a service.

`-e` *service* -m *member*

Enable a service on a specific member.

`-l`

Lock the service states.

`-r` *service* -m *member*

Relocate a service to a specific member.

`-q`

Operate quietly.

`-R`

Restart a service.

`-s`

Stop a service

`-u`

Unlock the service states.

`-v`

Displays information about the current version of `clusvcadm`.

Refer to the `clusvcadm`(8) man page for more information.

F.4. Using the `clufence` Utility

If power switches are used in the cluster hardware configuration, run the `clufence` utility on each cluster system to ensure that it can remotely power-cycle the other cluster members.

If the command succeeds, run the `shutil -p` command on both cluster systems to display a summary of the header data structure for the quorum partitions. If the output is different on the systems, the quorum partitions do not point to the same devices on both systems. Check to make sure that the raw devices exist and are correctly specified in the `/etc/sysconfig/rawdevices` file. See Section 2.4.4.3 *Configuring Shared Cluster Partitions* for more information.

If either network- or serial-attached power switches are employed in the cluster hardware configuration, install the cluster software and invoke the `clufence` command to test the power switches. Invoke the command on each cluster system to ensure that it can remotely power-cycle the other cluster members. If testing is successful, then the cluster can be started.

The `clufence` command can accurately test a power switch. The format of the `clufence` command is as follows:

```
usage: clufence [-d] [-[furs] <member>]
  -d              Turn on debugging
  -f <member>      Fence (power off) <member>
  -u <member>      Unfence (power on) <member>
  -r <member>      Reboot (power cycle) <member>
  -s <member>      Check status of all switches controlling <member>
```

When testing power switches, the first step is to ensure that each cluster member can successfully communicate with its attached power switch. The following example of the `clufence` command output shows that the cluster member is able to communicate with its power switch:

```
[23750] info: STONITH: baytech at 192.168.1.31, port 1 controls clu2
[23750] info: STONITH: baytech at 192.168.1.31, port 2 controls clu3
[23750] info: STONITH: wti_nps at 192.168.1.29, port clu4 controls clu4
[23750] info: STONITH: wti_nps at 192.168.1.29, port clu5 controls clu5
```

Any errors in the output could be indicative of the following types of problems:

- For serial attached power switches:
  - Verify that the device special file for the remote power switch connection serial port (for example, `/dev/ttyS0`) is specified correctly in the cluster database, as established via **Cluster Configuration Tool**. If necessary, use a terminal emulation package such as **minicom** to test if the cluster system can access the serial port.
  - Ensure that a non-cluster program (for example, a `getty` program) is not using the serial port for the remote power switch connection. You can use the `lsof` command to perform this task.
  - Check that the cable connection to the remote power switch is correct. Verify that the correct type of cable is used (for example, an RPS-10 power switch requires a null modem cable), and that all connections are securely fastened.
  - Verify that any physical dip switches or rotary switches on the power switch are set properly.

- For network based power switches:
  - Verify that the network connection to network-based switches is operational. Most switches have a link light that indicates connectivity.
  - It should be possible to `ping` the network switch; if not, then the switch may not be properly configured for its network parameters.
  - Verify that the correct password and login name (depending on switch type) have been specified in the cluster configuration database (as established by running **Cluster Configuration Tool**). A useful diagnostic approach is to verify Telnet access to the network switch using the same parameters as specified in the cluster configuration.

After successfully verifying communication with the switch, attempt to power cycle the other cluster member. Prior to doing this, it is recommended to verify that the other cluster member is not actively performing any important functions (such as serving cluster services to active clients). By executing the following command :

```
clufence -r clu3
```

The following depicts a successful power cycle operation:

```
Successfully power cycled host clu3.
```

Index

Symbols

/etc/hosts
  editing, 18
/etc/sysconfig/ha/lvs.cf file, 120
/etc/sysconfig/rawdevices
  editing the file, 41

A

acknowledgments, i
activating your subscription, vii
active router
  (see LVS clustering)
Apache HTTP Server
  httpd.conf, 94
  setting up service, 93
availability and data integrity table, 7

B

backup router
  (see LVS clustering)

C

channel bonding
  (see Ethernet bonding)
chkconfig, 121
cluster
    (see cluster types)
  administration, 97
  checking the configuration, 54, 180
  daemons, 42
  diagnosing and correcting problems, 102
  disabling the cluster software, 102
  displaying status, 97
  name, changing, 102
  using Red Hat Cluster Manager with Piranha, 151
cluster administration, 97
  backing up the cluster database, 100
  changing the cluster name, 102
  diagnosing and correcting problems in a cluster, 102
  disabling the cluster software, 102
  displaying cluster and service status, 97
  modifying cluster event logging, 101
  modifying the cluster configuration, 100
  restoring the cluster database, 100
  starting and stopping the cluster software, 99
  updating the cluster software, 101
cluster command-line utilities, 179
cluster communication mechanisms, 169
cluster configuration, 54
  minimum
    example, 7
  modifying, 100
  reloading, 41
  using the shutil utility, 180
Cluster Configuration Tool
  accessing, 38
  and Oracle service, 71
cluster daemons, 42
  membership daemon, 42
  quorum daemon, 43
  service manager daemon, 45
cluster database
  backing up, 100
  restoring, 100
cluster event logging
  daemons
    cluhbd, 101
    clupowerd, 101
    cluquorumd, 101
    clusvcmgrd, 101
  severity levels, 101
cluster hardware
  connecting, 21
  power controllers, 10
  setting up, 21
cluster hardware tables, 11
cluster member hardware table, 11
cluster service
  displaying status, 97
cluster services
  active-active NFS configuration, 83
  administration, 59
  Apache HTTP Server, setting up, 93
    httpd.conf, 94
  configuration, 51, 59
  configuring service disk storage, 61
  deleting a service, 64
  disabling a service, 63
  displaying a service configuration, 62
  enabling a service, 63
  gathering service information, 59
  handling a service that fails to start, 64
  modifying a service, 63
  MySQL service, setting up, 73
  NFS caveats, 82
  NFS client access, 82
  NFS Druid, 78
  NFS server requirements, 77
  NFS service, setting up, 77
  Oracle service, setting up, 67
  Oracle, tuning, 72
  relocating a service, 64

Samba Druid, 87
Samba operating model, 86
Samba server requirements, 85
Samba service, setting up, 85
scripts, creating, 61
smb.conf.sharename file fields, 90
verifying application software and service scripts, 62
cluster software
command-line utilities, 179
disabling, 102
installation and configuration, 35
steps for installing and initializing, 35
with rpm, 36
with the Package Management Tool, 35
starting and stopping, 99
steps for installing and initializing, 35
updating, 101
version display, 56
cluster software installation and configuration, 35
cluster types
compute-clustering
Beowulf, 109
definition of, 109
high-availability clustering, 109
(see also Red Hat Cluster Manager)
definition of, 109
load-balance clustering, 109
(see also LVS clustering)
definition of, 109
overview of, 109
clusvcadm
using, 180
common cluster behaviors
2-4 Member cluster, IP tie-breaker, 173
3-5 Member Cluster, 173
miscellaneous, 175
service daemons, 174
two member disk-based tie-breaker, 172
common cluster behaviors table, 170
components
of LVS cluster, 119
compute-clustering
(see cluster types)
configuration
Red Hat Enterprise Linux, 17
configuration file
reloading, 41
configuring a service, 59
console startup messages
displaying, 20
console switch, 9
setting up, 17
console switch hardware table, 15
conventions
document, iv

D

daemons
clulockd
configuring, 42
clumembd
configuring, 42
cluquorumd
configuring, 42
clurmtabd
configuring, 42
clusvcmgrd
configuring, 42
databases
MySQL
setting up service, 73
Oracle
oraclescript example, 68
setting up, 67
startdb script example, 68
stopdb script example, 69
tuning, 72
using Cluster Configuration Tool with, 71
deleting a service, 64
diagnosing and correcting problems in a cluster
table, 103
disabling a service, 63
disk storage
configuring service disk storage, 61
displaying a service configuration, 62
displaying console startup messages, 20
displaying devices configured in the kernel, 21

E

enabling a service, 63
Ethernet channel bonding
configuring, 22
event logging
modifying, 101
syslog
configuring, 57
examples
minimum cluster configuration, 7
NFS Druid, 78
no single point of failure configuration, 8
oracle script, 68
Samba Druid, 87
startdb script, 68
stopdb script, 69
using Cluster Configuration Tool to add an Oracle service, 71
ext3, 33

F

failover and recover scenarios, 170
feedback, vii
file services
  NFS
    active-active configuration, 83
    caveats, 82
    client access, 82
    NFS Druid, 78
    server requirements, 77
    setting up service, 77
  Samba
    operating model, 86
    Samba Druid, 87
    server requirements, 85
    setting up, 85
file systems
  creating, 33
FTP, clustering, 130
  (see also LVS clustering)

G

GFS
  cluster archive
    configuration, 161
  configuration
    with ccs_tool, 161
GFS Setup Druid, 157
  and CCS files, 158
  cluster name, 157
  configuration, 157
  configuring cluster members, 159
  fence device configuration, 160
  GULM settings, 158
  member configuration, 160
  post-configuration, 161
  saving configuration, 161
Grand Unified Lock Manager
  (see GULM)
GULM, 158
  configuration with GFS Setup Druid, 158

H

handling a service that fails to start, 64
hardware
  installing basic cluster hardware, 15
hardware configuration
  availability considerations, 6
  choosing a configuration, 5
  cost restrictions, 6
  data integrity under all failure conditions, 6
  minimum, 6
  optional hardware, 9
  performance considerations, 5
  shared storage requirements, 6
hardware information, supplementary, 163
hardware installation
  operating system configuration, 5
hardware watchdog timer
  configuring, 165
hardware watchdog timers, 165
high-availability clustering
  (see cluster types)
how to use this manual, iii
HTTP services
  Apache HTTP Server
    httpd.conf, 94
    setting up, 93

I

installation
  Red Hat Enterprise Linux, 17
installing basic cluster hardware, 15
installing the basic cluster hardware, 15
introduction, iii
  how to use this manual, iii
  other Red Hat Enterprise Linux manuals, iii
iptables, 121
ipvsadm program, 120

J

job scheduling, LVS, 114

K

kernel
  decreasing kernel boot timeout limit, 19
  displaying configured devices, 21
Kernel Boot Timeout Limit
  decreasing, 19
Kimberlite, i
KVM (keyboard, video, mouse) switch, 10

L

least connections
(see job scheduling, LVS)
Linux Virtual Server
(see LVS clustering)
load-balance clustering
(see cluster types)
low voltage differential (LVD), 167
LVS
/etc/sysconfig/ha/lvs.cf file, 120
components of, 119
daemon, 120
date replication, real servers, 113
definition of, 109
initial configuration, 121
ipvsadm program, 120
job scheduling, 114
lvs daemon, 120
LVS routers
configuring services, 121
necessary services, 121
primary node, 121
multi-port services, 129
FTP, 130
nanny daemon, 120
NAT routing
enabling, 127
requirements, hardware, 125
requirements, network, 125
requirements, software, 125
overview of, 110, 111
packet forwarding, 124
Piranha Configuration Tool, 120
pulse daemon, 119
real servers, 110
routing methods
NAT, 116
routing prerequisites, 125
scheduling, job, 114
send_arp program, 120
shared data, 113
starting the cluster, 148
synchronizing configuration files, 147
three tiered
Red Hat Cluster Manager, 113
using Red Hat Cluster Manager with Piranha, 151
lvs daemon, 120

M

member status table, 98
members
setting up, 15
membership daemon, 42
minimum cluster configuration example, 7
minimum hardware configuration, 6
Mission Critical Linux, Inc., i
mkfs, 33
multi-port services, clustering, 129
(see also LVS clustering)
MySQL
setting up service, 73

N

nanny daemon, 120
NAT
enabling, 127
routing methods, LVS, 116
network address translation
(see NAT)
network hardware table, 13
network hub, 9
network switch, 9
NFS
active-active configuration, 83
caveats, 82
client access, 82
NFS Druid, 78
server requirements, 77
setting up service, 77
NMI watchdog timer
enabling, 164
no single point of failure configuration, 8
Non-Maskable Interrupt (NMI) watchdog timers, 164

O

operating system configuration
hardware installation, 5
Oracle
adding an Oracle service, 71
oracle script example, 68
setting up service, 67
startdb script example, 68
stopdb script example, 69
tuning service, 72

P

packet forwarding, 124
  (see also LVS clustering)
Parallel SCSI
  requirements, 26
parted
  creating disk partitions, 31
partitioning disks, 31
Piranha
  using Red Hat Cluster Manager with, 151
Piranha Configuration Tool, 120
  CONTROL/MONITORING, 134
  EDIT MONITORING SCRIPTS Subsection, 145
  GLOBAL SETTINGS, 135
  limiting access to, 123, 123
  login panel, 133
  necessary software, 133
  overview of, 133
  REAL SERVER subsection, 142
  REDUNDANCY, 137
  setting a password, 122
  VIRTUAL SERVER subsection
    Firewall Mark, 141
    Persistence, 142
    Scheduling, 141
    Virtual IP Address, 140
  VIRTUAL SERVER subsection, 139
  VIRTUAL SERVERS, 139
piranha-gui service, 121
piranha-passwd, 122
point-to-point Ethernet connection hardware table, 14
power controller connection, configuring, 48
power controllers, 10
  network-attached, 10
  serial-attached, 10
  watchdog timers, 10
    hardware-based, 10
    software-based, 10
power switch, 48
  (see also power controller)
power switch hardware table, 12
power switches
  configuring, 23
  hardware watchdog timers
    configuring, 165
  NMI watchdog timers
    enabling, 164
  setting up, 163
    watchdog, 163
  software watchdog timers
    configuration, 163
  testing, 55, 181
  troubleshooting, 55, 182
pulse daemon, 119
pulse service, 121

Q

quorum daemon, 43
quorum partitions
    (see shared cluster partitions)
  testing, 180

R

raw, 32
raw devices
  creating, 32
rawdevices
  editing the file, 41
real servers
    (see LVS clustering)
  configuring services, 124
Red Hat Cluster Manager, 109
  and Piranha, 151
  command-line utilities, 179
  with Red Hat GFS, 157
Red Hat Cluster Suite, 35
  installation, 35
    with rpm, 36
    with the Package Management Tool, 35
Red Hat Enterprise Linux, 109
  installation and configuration, 17
  overview of, 109
Red Hat GFS
  and Red Hat Cluster Suite, 157
Red Hat GFS with Red Hat Cluster Suite
  changes to Red Hat Cluster Suite, 154
  installation scenarios, 154
  terminology, 153
  using, 153
  using GFS Setup Druid, 157
registering your subscription, vii
relocating a service, 64
round robin
  (see job scheduling, LVS)
routing
  prerequisites for LVS, 125

S

Samba
operating model, 86
Samba Druid, 87
server requirements, 85
setting up service, 85
smb.conf.sharename file fields, 90
scheduling, job (LVS), 114
scripts
creating service scripts, 61
oracle script example, 68
startdb script example, 68
stopdb script example, 69
verifying application software and service scripts, 62
SCSI bus configuration requirements, 165
SCSI bus length, 167
SCSI bus termination, 166
SCSI identification numbers, 167
security
Piranha Configuration Tool, 123
send_arp program, 120
service configuration, 59
service property and resource information table, 60
service status table, 98
services, 51
(see also adding to the cluster configuration)
shared cluster partitions
configuring, 30
requirements, 30
shared disk storage
configuring, 26
shared disk storage hardware table, 13
shared partitions
testing, 54
shared storage, 26
shared storage requirements, 6
single-initiator fibre channel interconnect
setting up, 28
single-initiator SCSI bus
setting up, 27
software information, supplementary, 169
software watchdog timers, 163
sshd service, 121
subscription registration, vii
synchronizing configuration files, 147
syslog, 57
syslog event logging
configuring, 57
syslogd, 57
System V init, 99

T

tables
availability and data integrity, 7
cluster hardware, 11
cluster member hardware, 11
common cluster behaviors, 170
console switch hardware, 15
diagnosing and correcting problems in a cluster, 103
installing the basic cluster hardware, 15
member status, 98
minimum cluster configuration components, 7
network hardware, 13
no single point of failure configuration, 8
point-to-point Ethernet connection hardware, 14
power controller connection, configuring, 48
power switch hardware, 12
service property and resource information, 60
service status, 98
shared disk storage hardware, 13
UPS system hardware, 14
terminal server, 10
testing
power switches, 55, 181
quorum partitions, 180
shared partitions, 54
troubleshooting
diagnosing and correcting problems in a cluster, 102
failover and recover scenarios, 170
power switch testing, 55, 182
table, 103

U

UPS system hardware table, 14
UPS systems
configuring, 24
using Red Hat GFS with Red Hat Cluster Suite, 153
utilities
clufence, 181
clusvcadm, 180
redhat-config-cluster-cmd, 179
shutil, 180

W

watchdog timers
  hardware
    configuring, 165
  hardware-based, 10
  NMI
    enabling, 164
  setting up, 163
  software, 163
    configuration, 163
  software-based, 10
weighted least connections
  (see job scheduling, LVS)
weighted round robin
  (see job scheduling, LVS)

Colophon

The manuals are written in DocBook SGML v4.1 format. The HTML and PDF formats are produced using custom DSSSL stylesheets and custom jade wrapper scripts. The DocBook SGML files are written in **Emacs** with the help of PSGML mode.

Garrett LeSage created the admonition graphics (note, tip, important, caution, and warning). They may be freely redistributed with the Red Hat documentation.

The Red Hat Product Documentation Team consists of the following people:

Sandra A. Moore — Primary Writer/Maintainer of the *Red Hat Enterprise Linux Installation Guide for x86, Itanium™, AMD64, and Intel® Extended Memory 64 Technology (Intel® EM64T)*; Primary Writer/Maintainer of the *Red Hat Enterprise Linux Installation Guide for the IBM® POWER Architecture*; Primary Writer/Maintainer of the *Red Hat Enterprise Linux Installation Guide for the IBM® S/390® and IBM® eServer™ zSeries® Architectures*

John Ha — Primary Writer/Maintainer of the *Red Hat Cluster Suite Configuring and Managing a Cluster*; Co-writer/Co-maintainer of the *Red Hat Enterprise Linux Security Guide*; Maintainer of custom DocBook stylesheets and scripts

Edward C. Bailey — Primary Writer/Maintainer of the *Red Hat Enterprise Linux Introduction to System Administration*; Primary Writer/Maintainer of the *Release Notes*; Contributing Writer to the *Red Hat Enterprise Linux Installation Guide for x86, Itanium™, AMD64, and Intel® Extended Memory 64 Technology (Intel® EM64T)*

Karsten Wade — Primary Writer/Maintainer of the *Red Hat SELinux Application Development Guide*; Primary Writer/Maintainer of the *Red Hat SELinux Policy Writing Guide*

Andrius Benokraitis — Primary Writer/Maintainer of the *Red Hat Enterprise Linux Reference Guide*; Co-writer/Co-maintainer of the *Red Hat Enterprise Linux Security Guide*; Contributing Writer to the *Red Hat Enterprise Linux System Administration Guide*

Paul Kennedy — Primary Writer/Maintainer of the *Red Hat GFS Administrator's Guide*; Contributing Writer to the *Red Hat Cluster Suite Configuring and Managing a Cluster*

Mark Johnson — Primary Writer/Maintainer of the *Red Hat Enterprise Linux Desktop Configuration and Administration Guide*

Melissa Goldin — Primary Writer/Maintainer of the *Red Hat Enterprise Linux Step By Step Guide*

The Red Hat Localization Team consists of the following people:

Amanpreet Singh Alam — Punjabi translations

Jean-Paul Aubry — French translations

David Barzilay — Brazilian Portuguese translations

Runa Bhattacharjee — Bengali translations

Chester Cheng — Traditional Chinese translations

Verena Fuehrer — German translations

Kiyoto Hashida — Japanese translations

N. Jayaradha — Tamil translations

Michelle Jiyeen Kim — Korean translations

Yelitza Louze — Spanish translations

Noriko Mizumoto — Japanese translations

Ankitkumar Rameshchandra Patel — Gujarati translations

Rajesh Ranjan — Hindi translations

Nadine Richter — German translations
Audrey Simons — French translations
Francesco Valente — Italian translations
Sarah Wang — Simplified Chinese translations
Ben Hung-Pin Wu — Traditional Chinese translations